Ph.D. Thesis
Doctor of Philosophy

DTU Physics
Department of Physics

Machine Learning in 4D Seismic Data Analysis

Deep Neural Networks in Geophysics

Jesper Sören Dramsch

Kongens Lyngby 2019

DTU

DTU Physics
Department of Physics
Technical University of Denmark

Fysikvej
Building 311
2800 Kgs. Lyngby, Lyngby
info@fysik.dtu.dk
Tel.: +45 4525 3344
https://www.fysik.dtu.dk

Abstract

Machine Learning provides an important tool for the modelling and analysis of geoscientific data. I have placed recent developments in deep learning into the greater context of machine learning by prefacing my work with a comprehensive history of machine learning and have reviewed the approaches and challenges of the use of machine learning in geoscience specifically. I have compiled a synopsis of the following chapters that contain topical peer-reviewed papers.

This thesis follows the full data science workflow, starting with familiarization of the data domain in one published journal article, one published conference paper and one published workshop paper. It is followed by groundwork on machine learning and data processing on 4D seismic data in two submitted journal articles, one published conference paper and one published workshop paper. Based on this groundwork, I present a method for 4D seismic inversion in two published workshop papers. Finally, I present a novel unsupervised 3D time-shift extraction method for 4D seismic in one submitted journal paper.

The aim of this thesis is to apply recent developments in computer vision systems and neural networks to physical data, particularly 4D seismic analysis. Neural networks are a type of machine learning that has made significant contributions to modern artificial intelligence and automatization. The applicability of neural networks for their capability of being a universal function approximator was recognized within geophysics from an early stage. With the deep learning boom, neural networks have experienced a renaissance in geoscience applications, particularly automatic seismic interpretation, inversion processes and sequence modelling.

The data for this thesis was acquired in the Danish North Sea, which contains chalk deposits, a sedimentologically distinct feature in the seismic data. The hydrocarbon-reservoir within the chalk has been subject to well-log analysis and core sampling in addition to seismic interpretation and 4D seismic analysis. During familiarization with the data, a new method to delineate chalk sediment in back-scatter scanning electron microscopy is introduced. Moreover, core fracture patterns, well imaging and seismic data are analyzed and compiled into a new workflow to ensure alignment of local and regional stress regimes.

Considering the wide interest in machine learning, my research investigates the following assumptions. The first paper shows that using pre-trained neural networks on natural images can reduce the data necessary for transfer learning to geoscience problems. I go on to analyze aliasing in neural networks and built a framework for complex-valued convolutional and dense neural networks to test the assumption that phase information can be implicitly learnt by real-valued neural networks. I further show that complex-

valued convolutions can stabilize training and data compression on non-stationary physical data.

During the external research stay, a collaboration with an expert on Bayesian inversion for pressure-saturation inversion from 4D seismic amplitude difference maps resulted in a novel deep dense sample-based encoder-decoder network that learns the inversion process. The network contains a low-assumption physical basis (AVO) and learns the residual for the inversion process. My work shows that transfer from simulation data to field data is possible.

Finally, an unsupervised method is devised to extract 3D time-shifts from two 4D seismic cubes. The network extracts these 3D time-shifts with the inclusion of uncertainty measures. Commonly, time-shifts are extracted in 1D, due to processing speed, computational cost and poor performance of 3D methods. Within the training loop, the stationary velocity field is numerically integrated to obtain a diffeomorphic warp field that constrains the topology in a geologically consistent manner. The unsupervised implementation of the network structure ensures that biases from other time-shift extraction methods are not implicitly included in the network.

Overall, this thesis presents two new methods for the application of deep learning in 4D seismic analysis. Moreover, this thesis dives into information-theoretical implications of neural networks for non-stationary data such as seismic, and presents several ways to apply deep learning in a data regime, where ground truth is expensive, sparse, and sometimes impossible to obtain. These include transfer learning of pre-trained networks and transfer from simulation to field data. Additionally, we show an application of unsupervised learning, by devising a way of behaviour for the network to follow instead of supplying ground truth labels. Moreover, this results in a way to increase trust in the system, by limiting the extraction process to the deep learning system and performing well-defined operations within the network to automate the training, therefore, making the process transparent.

Dansk Resumé

Maskinlæring ('machine learning') er et vigtigt redskab til modellering og analyse af geovidenskabelige data. Jeg har sat den seneste udvikling inden for dyb læring ('deep learning') ind i en større sammenhæng via et forord, der gennemgår maskinlæringens historie, samt metoderne til og udfordringerne ved brug af maskinlæring, specifikt inden for geovidenskab. Afhandlingen består af en synopsis af de publicerede og indsendte peer-reviewed afhandlinger i de derefter følgende kapitler.

Denne afhandling starter med at give overblik over af data via en publiceret tidsskriftsartikel, en publiceret konferenceartikel og en publiceret workshopartikel. Dernæst følger forarbejdet til maskinlæring og databehandling af 4D seismiske data i to indsendte tidsskriftsartikler, en publiceret konferenceartikel og en publiceret workshopartikel. På baggrund af dette forarbejde, præsenterer jeg en metode til 4D seismisk inversion i to publicerede workshopartikler. Endelig præsenterer jeg en ny, 3D ekstraktionsmetode med tidsforskydning ('time shift') for 4D seismiske data med brug af unsupervised learning i en indsendt tidsskriftartikel.

Formålet med denne afhandling er, at anvende den seneste udvikling inden for systemer for computer vision og neurale netværk for fysiske data, især 4D seismisk analyse. Neurale netværk er en type maskinlæring, der har bidraget stort inden for moderne, kunstig intelligens og automatisering. Det blev på et tidligt tidspunkt anerkendt inden for geofysik, at neurale netværk var anvendelige. Med fremgangen inden for dyb læring, har neurale netværk oplevet en renæssance inden for geovidenskabelige anvendelser, især automatisk seismisk tolkning, inversionsprocesser og sekvensmodellering.

Data til denne afhandling, er indhentet fra den danske del af Nordsøen, der indeholder kridtaflejringer, hvilket er et sedimentologisk distinkt udtryk i de seismiske data. Kulbrinte-reservoiret i kridtet har været genstand for borehulsanalyse og kerneprøvetagning samt seismisk tolkning og 4D seismisk analyse. I forbindelse med arbejdet blev en ny metode til afgrænsning af kridtsediment i scanning-elektronmikroskopi med tilbagespredning ('BSEM') introduceret. Desuden blev kernefrakturmønstre, borehulsbilleder og seismiske data analyseret og samlet i en ny arbejdsgang for at justere lokale- og regionale stressregimer.

Set i lyset af den store interesse for maskinlæring, undersøger min forskning flere områder. Den første artikel viser, at brugen af trænede neurale netværk på billeder, kan reducere de data, der er nødvendige for at overføre læring til geovidenskabelige problemer. Jeg fortsætter med at analysere aliasing i neurale netværk og udvikler et computerprogram til at bygge neurale netværk, som bruger komplekse tal. Jeg sammenligner dette med netværk, som kun bruger ikke-komplekse tal, for at teste gendannelse

af data og datakomprimering af ikke-stationære fysiske data.

Under det eksterne forskningsophold kom et samarbejde i stand, der udarbejder et nyt dybt, tæt prøvebaseret indkoder-dekoder-netværk, der lærer inversionsprocesser. Netværket indeholder et fysisk grundlag, for selv at lære resten af inversionsprocessen. Mit arbejde viser, at overførsel fra indhentet data til simulationsdata er muligt.

Endelig blev der udviklet en 'unsupervised' metode, til at udregne 3D-tidsforskydninger fra to 4D seismiske kuber. På grund af de beregningsmæssige omkostninger og dårlige kvalitet, bliver disse normalt kun beregnet i 1D. Det neurale netværket beregner 3D tidsforskydningerne inklusiv usikkerhedsmålinger og er brugt på tre forskellige seismiske datasæt. Den 'unsupervised' implementering af netværksstrukturen sikrer, at bias fra andre tidsforskydnings ekstraktionsmetoder ikke implicit indgår i netværket.

Samlet set, præsenterer denne afhandling nye metoder inden for anvendelsen af dyb læring i 4D seismisk analyse. Desuden ser afhandlingen på de informationsteoretiske konsekvenser af neurale netværk til ikke-stationære data, såsom seismik, og præsenterer flere måder at anvende dyb læring på i et dataregime, hvor faktiske data er dyre at indsamle, af dårlig kvalitet og nogle gange umulige at fremskaffe. Disse inkluderer overførsel af læring i præ-trænede ('pre-trained') netværk og overførsel fra simulationsdata til målt/indsamlet data.

Preface

This dissertation is presented by

Jesper Sören Dramsch

to the

Department of Physics

in partial fulfilment of the requirements for the degree of

Doctor of Philosophy (Ph.D.)

at the

Technical University of Denmark

Kongens Lyngby, November 14th, 2019

Ph.D. Thesis

University:	Technical University of Denmark (DTU)
Department:	Department of Physics / Centre for Oil and Gas – DTU
Author:	Jesper Sören Dramsch
Title:	Machine Learning in 4D Seismic Data Analysis Deep Neural Networks in Geophysics
Principal Advisor:	Mikael Lüthje
Co-Advisor:	Anders Nymark Christensen (DTU Compute)
External Advisor:	Colin MacBeth (Heriot-Watt University, UK)
Submitted:	2019-11-14

Acknowledgements

This thesis would not exist, without the support of my peers.

My sincere gratitude to my supervisor Mikael Lüthje for his continuous support, supervision and guidance through the perilous cliffs of pursuing a Ph.D. in a new and ever-evolving academic center. The open discussions and trust in my ability allowed me to thrive during this period. My appreciation extends to Colin MacBeth, who in the position as my external co-supervisor welcomed me to Edinburgh and provided further guidance and insight into the practical workings of 4D seismics. I am deeply indebted to my internal co-supervisor Anders Nymark Christensen, who provided valuable insight into the statistical working of machine learning and kept my weights and biases in check. Thank you for inspiring me to achieve more than I ever thought possible.

To the friends we had and made along the way! Kirstie Wright and Anna Clark you kept me sane from day to day and made Scotland feel home. Thank you. Robert Leckenby, Tim Albrecht, Bettina Schmidt, Matthias Schneider, Manuela Köllner, Clara Dabrock, Brian Burnham and Florian Smit, you were always there and I appreciate you for it. Marie-Daphne Mangriotis thank you for welcoming me to ETLP and the great conversations. Furthermore, I would like to thank Antony Hallam and Gustavo Corte for great discussion and peership.

My thanks go out to the Software Underground community, for keeping the spirit of sharing and collaboration. Especially, Matt Hall for the leadership and trust. To Lukas Mosser, for always encouraging and inspiring me to strive for more.

I would like to thank my colleagues at the DHRTC, particularly Tala Maria Aabø for being a fantastic co-conspirator in the early days and Florian Smit for welcoming me into this new environment. Furthermore, I'd like to thank Frédéric Amour, Charlotte Lind Laurentzius, Anne Lysgaard and Helle Baumann for being a pleasure to work with.

I want to thank part of the Open Source community and in particular the scientific Python community, without these tools this thesis would be much less substantial.

Difficulties strengthen the mind,
as labor does the body.

SENECA THE YOUNGER

Publication List

Journal Articles

Aabø, T. M., **J. S. Dramsch**, C. L. Würtzen, S. Seyum, F. Amour, M. Welch, and M. Lüthje (2020). "An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field". In: *Marine and Petroleum Geology* 112. Published, Chapter 4. ISSN: 0264-8172. DOI: https://doi.org/10.1016/j.marpetgeo.2019.104065. URL: http://www.sciencedirect.com/science/article/pii/S026481721930501X.

Corte, G., **J. S. Dramsch**, C. MacBeth, and H. Amini (2019). "Exploring training possibilities in a Deep Neural network application for Inverting 4D seismic maps to changes in pressure and saturation". In: *TBD*. In Preparation, Not Included.

Dramsch, **J. S.**, A. N. Christensen, and M. Lüthje (2019a). "Let's do the Time Warp again! – Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data". In: *Geophysics*. In Review, Chapter 5.

Dramsch, **J. S.**, A. N. Christensen, C. MacBeth, and M. Lüthje (2019c). "Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks". In: *IEEE Transactions in Geoscience and Remote Sensing*. Submitted, Chapter 7.

Dramsch, **J. S.**, G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019i). "Physics-based deep neural encoders-decoder network for 4D seismic pressure-saturation inversion on North Sea data". In: *TBD*. In Preparation, Not Included.

Dramsch, **J. S.**, M. Lüthje, and A. N. Christensen (2019j). "Complex-valued neural networks for machine learning on non-stationary physical data". In: *Computers & Geoscience*. Submitted, Chapter 5.

Peer-Reviewed Conference Proceedings

Dramsch, **J. S.** and M. Lüthje (2018d). "Deep-learning seismic facies on state-of-the-art CNN architectures". In: *SEG Technical Program Expanded Abstracts 2018*. Published, Chapter 5. Society of Exploration Geophysicists, pp. 2036–2040. DOI: 10.1190/segam2018-2996783.1. URL: https://doi.org/10.1190/segam2018-2996783.1.

Mosser, L., W. Kimman, **J. S. Dramsch**, S. Purves, A. De la Fuente Briceño, and G. Ganssle (2018a). "Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks". In: *80th EAGE Conference and*

Exhibition 2018. Not Included. EAGE. DOI: 10.3997/2214-4609.201800734. URL: https://doi.org/10.3997/2214-4609.201800734.

Aabø, T. M., **J. S. Dramsch**, M. Welch, and M. Lüthje (2017a). "Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea". In: *79th EAGE Conference and Exhibition 2017*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201701283. URL: https://doi.org/10.3997/2214-4609.201701283.

Peer-Reviewed Workshop Proceedings

Dramsch, **J. S.**, G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019e). "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data". In: *Second EAGE Workshop Practical Reservoir Monitoring 2019*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201900028.

Dramsch, **J. S.**, G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019h). "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201901967. URL: https://doi.org/10.3997/2214-4609.201901967.

Dramsch, **J. S.**, F. Amour, and M. Lüthje (2018a). "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201803014. URL: https://doi.org/10.3997/2214-4609.201803014.

Dramsch, **J. S.** and M. Lüthje (2018e). "Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 5. EAGE. DOI: 10.3997/2214-4609.201803020. URL: https://doi.org/10.3997/2214-4609.201803020.

Open Source Software List

Open Source Packages

Dramsch, **J. S.** and Contributors (2019d). *Complex-Valued Neural Networks in Keras with Tensorflow*. Open-Source Software. DOI: 10.6084/m9.figshare.9783773. URL: https://github.com/JesperDramsch/keras-complex.

Reproducible Code

Dramsch, **J. S.** (2020). *Reproducible Code: Dynamic Timewarping Tutorial – Geophysics*. URL: https://github.com/JesperDramsch/dtw-tutorial.

Dramsch, **J. S.** (2019c). *Reproducible Code: Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks*. URL: https://github.com/JesperDramsch/voxelmorph-seismic.

Dramsch, **J. S.** (2018e). *Reproducible Code: Deep-learning seismic facies on state-of-the-art CNN architectures*. DOI: 10.6084/m9.figshare.7227545. URL: https://github.com/JesperDramsch/segam18.

Contributions to Free Open Source Software

Tensorflow: Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: http://tensorflow.org/

Pandas: W. McKinney et al. (2010). "Data structures for statistical computing in python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX, pp. 51–56

Open Geoscience Awesome List: J. S. Dramsch and Contributors (2018b). *Awesome Open Geoscience*. Maintainer. URL: https://github.com/softwareunderground/awesome-open-geoscience

Bruges: *Bruges: Bag of really useful geophysical equations and stuff* (2016). URL: https://github.com/agile-geoscience/bruges

Scikit-Learn: F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830

Presentation List

Conference Presentation

Dramsch, **J. S.** and M. Lüthje (2018d). "Deep-learning seismic facies on state-of-the-art CNN architectures". In: *SEG Technical Program Expanded Abstracts 2018*. Published, Chapter 5. Society of Exploration Geophysicists, pp. 2036–2040. DOI: 10.1190/segam2018-2996783.1. URL: https://doi.org/10.1190/segam2018-2996783.1.

Mosser, L., W. Kimman, **J. S. Dramsch**, S. Purves, A. De la Fuente Briceño, and G. Ganssle (2018a). "Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks". In: *80th EAGE Conference and Exhibition 2018*. Not Included. EAGE. DOI: 10.3997/2214-4609.201800734. URL: https://doi.org/10.3997/2214-4609.201800734.

Workshop Presentation

Dramsch, **J. S.**, G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019e). "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data". In: *Second EAGE Workshop Practical Reservoir Monitoring 2019*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201900028.

Dramsch, **J. S.**, G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019h). "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201901967. URL: https://doi.org/10.3997/2214-4609.201901967.

Dramsch, **J. S.**, F. Amour, and M. Lüthje (2018a). "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201803014. URL: https://doi.org/10.3997/2214-4609.201803014.

Workshop Poster

Dramsch, **J. S.** and M. Lüthje (2018e). "Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 5. EAGE. DOI: 10.3997/2214-4609.201803020. URL: https://doi.org/10.3997/2214-4609.201803020.

Other Presentations

Dramsch, **J. S.**, G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019g). *Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data*. ETLP Sponsor Meeting 2019.

Dramsch, **J. S.** (2018c). *KFold in Deep Learning*. Lightning Talk – EuroScipy 2018. DOI: 10.6084/m9.figshare.7035908. URL: https://doi.org/10.6084/m9.figshare.7035908.

Other Posters

Dramsch, **J. S.**, A. N. Christensen, and M. Lüthje (2019b). *Physics and Deep Learning - Incorporating prior knowledge in deep neural networks*. DOI: 10.6084/m9.figshare.8217518.v1.

Dramsch, **J. S.** and M. Lüthje (2018c). *Deep Learning: From Cats to 4D Seismic - Reducing cycle time and model training cost in asset management*. Tech. rep. Danish Hydrocarbon Research and Technology Centre. DOI: 10.6084/m9.figshare.7422629.

Dramsch, **J. S.** (2017b). *Edge detection in 3D seismic data*. DHRTC PhD Day.

Invited Presentation

Dramsch, **J. S.** (2019a). *Cracking Open the Black Box – Making sense of Machine Learning and Neural Networks*. EAGE E-Lecture.

Dramsch, **J. S.** (2019b). *Making sense of AI for a career in a changing industry*. EAGE Annual Meeting 2019.

Dramsch, **J. S.** (2018a). *4D Seismics in Fracture Characterization – A machine learning perspective*. Company Talk - ConocoPhillips.

Dramsch, **J. S.** (2018b). *A practitioner's guide to deep learning in geophysical imaging*. FORCE Velocity Modeling Meeting. DOI: 10.6084/m9.figshare.7170299.v1.

Dramsch, **J. S.** (2018d). *Machine Learning Workshop*. Heriot-Watt University, ETLP.

MacBeth, C., R. Chassagne, and **J. S. Dramsch** (2018). *A guided discussion on machine learning for 4D QI*. ETLP Sponsor Meeting 2018.

Dramsch, **J. S.** (2017a). *Edge detection in 3D seismic*. DTU Vision Day.

Contents

CHAPTER 1

Introduction

This thesis explores machine learning in geoscience with a special focus on deep learning in 4D seismics. Recently, machine learning and neural networks in particular have made important impacts in many scientific disciplines, with geoscience exploring these new approaches as well. This study contributes to this body of emerging work in deep neural networks and computer vision systems for the modelling and analysis of geoscientific data. The main contribution being a physics-based neural architecture for pressure-saturation inversion and a novel algorithm for 3D time-shift extraction in 4D seismic.

The growing interest in machine learning sometimes overlooks the fact that machine learning as a concept was introduced in 1950. Geoscience and in particular geophysics has followed the innovation in artificial intelligence and especially neural networks closely. Early applications of neural networks include seismic processing and seismic inversion. Moreover, Gaussian processes were early introduced in geostatistics as kriging, being the primary application of Gaussian processes for a period of time. Deep learning becoming popular and particularly breakthroughs in computer vision have sparked interest in applying machine learning computer vision to seismic interpretation in the hopes for increased accuracy, reproducibility and automation.

In recent years, 4D seismic itself has made an impact in geoscience. The method enables imaging of changes in the subsurface. This is essential in hydrocarbon production, enabling extended production reducing the direct environmental footprint and ensuring resource safety. Moreover, it enables CO_2 sequestration monitoring for reservoir and seal integrity and applications including nuclear test treaty compliance, waste storage, and deep geothermal monitoring. 4D seismic matching has exposed deficits in 3D seismic processing, therefore furthered our understanding of amplitude-preserving and surface-consistent processing steps. Additionally, furthering our understanding of in-situ validation of geomechanical concepts and update of heterogeneous subsurface models.

The structure of this study is composed of a theoretical introduction into 4D seismic principles, followed by a thorough overview of the development of machine learning in general to provide context for a review of machine learning in geoscience. This leads into a discussion of challenges for machine learning in geoscience. The theoretical foundation serves as the basis for ten publications that are arranged into four topical chapters.

The first chapter comprised of three papers investigates image processing for improved seismic interpretation. This workflow was essential in analyzing the local to regional stress fields from fracture and fault expressions in 3D seismic data. Familiarization with the available data set proved to be valuable in the analysis of the 4D seismic data set and fine-tuning machine learning algorithms to the specific domain presented. Then unsupervised machine learning is applied to distinguish chalk sediments in back-

scatter electron microscopy, providing a machine learning solution for a normally manual and tedious task.

The second chapter comprised of four papers investigates fundamentals of signal processing for 4D seismic and neural networks. In which different metrics and constraints for dynamic time warping are explored, introducing a novel constraint for warping traces, significantly improving the alignment of 4D seismic traces in the base and monitor volume. Then aliasing and the impact of including phase information in neural networks is investigated. For the purpose of this study open source software was translated to the modern Tensorflow framework to enable building complex-valued convolutional neural networks. This chapter concludes in investigating transfer learning of pre-trained neural networks on natural images applied to seismic data, introducing a method to apply deep learning in label sparse environments.

The third chapter comprised of two papers introduces a deep neural network architecture for 4D quantitative pressure-saturation inversion. The regression model implements a layer that computes basic physical knowledge within the network architecture to stabilize the network. The physical knowledge encoded in the layer is the AVO gradient between the input seismic data. This data is passed into an variational encoder-decoder architecture. In this work we show that this network can be trained on simulation data and transferred to field data by applying Gaussian noise to the noise-free simulation input data to condition the network to accept noisy inputs from field data.

The fourth chapter comprised of a single paper introduces a robust method for 3D time shift extraction in 4D data. Time shifts in 4D data are commonly extracted in 1D due to computational cost and often poor performance of 3D methods. This method uses a deep learning system to extract the mapping of two seismic volumes without supplying a-priori time shift data, training self-supervised. Moreover, the method limits the neural network to the extraction of the stationary warp velocity field but leaves the warping to a non-learning 3D interpolation to increase transparency of the method. Additionally, the method supplies uncertainty values for the warp velocity. Constraining the possible 3D time shifts is important to ensure sensible results for the time shifts, as well as, the aligned monitor seismic. This is ensured by implementing a geologically intuitive constraint on the warp-field, namely a diffeomorphic mapping, which prohibits crossing or looping of reflectors after warping. This learning-based method can be trained in advance, providing timely results on unseen data, which is essential in 4D seismic analysis.

CHAPTER 2
Methods & Theory

This thesis applies Machine Learning methods to 4D seismic data. In this chapter I introduce 4D seismic concepts and the motivation to acquire and analyze 4D seismic data. I go on to introduce machine learning and review the development of machine learning in itself and in the field of geoscience. The focus on this thesis is on Neural Networks, particularly Deep Learnings to geophysical problems. Considering recent developments in computer vision, a focus on Convolutional Neural Networks, the developments and break-throughs of this type of Neural Network (NN) and the innovations that lead to the recent adoption of Machine Learning in geoscience are explored.

2.1 4D seismic

4D seismic is the analysis of seismic data that was acquired over the same location after some calendar time has passed. The repeated imaging of the same subsurface location, highlights changes in the subsurface that can lead to improved understanding of subsurface processes and fluid movement. E&P companies in particular have an interest in imaging hydrocarbon reservoirs (Johnston, 2013b), however 4D seismic imaging wide applications for subsurface characterization, such as observing volcanic activitiy (Londoño et al., 2018) or CO_2 sequestration monitoring (Arts et al., 2004).

The main applications of 4D seismic analysis according to Yilmaz (2003) and Johnston (2013a) include:

- Tracking fluid movement (steam, gas, and water)
- Monitoring pressure depletion and validating depletion plans
- Fault property estimation i.e. sealing or leaking faults
- Locating bypassed oil in heterogeneous reservoirs
- Validating and updating geological and reservoir-simulation models

4D seismic data analysis suffers from the superposition of multiple effects on the seismic imaging. These effects include changes in the acquisition equipment due to technological advances, changes in acquisition geometry (source-receiver mismatch), as well as physical changes in the subsurface (Yilmaz, 2003; Johnston, 2013b). These physical changes are in part due to fluid movement in the subsurface (Lumley, 1995),

as well as, changes in the geology due to compaction and expansion (Hatchell et al., 2005a). These geomechanical effects change the position of the reflectors, the thickness of stratigraphy and the physical properties such as density and wave velocity (Herwanger, 2015).

Succesfull 4D applications rely on careful acquisition planning, closely matching the mismatch of source (ΔS) and receiver (ΔR). This awareness has generally improved the repeatability of seismic acquisition, however, the Normalized Root Mean Squared Error (NRMS) remains to be an important measure of noise sources that deteriorate the 4D seismic analysis. Moreover, 4D seismic analysis has brought to light that some 3D seismic processing workflows are not as repeatable and amplitude-preserving as they were thought to be (Lumley, 2001). Modern processing flows include co-processing of the base and monitor seismic volumes with specialized tools to reduce differences from processing (Johnston, 2013a).

The standard analysis tool in 4D seismic interpretation are amplitude differences (Johnston, 2013b). Differences can stem from fluid movement or replacement and changes in the rock matrix due to compaction, temperature changes, and movement of injected CO_2 plumes. Additionally, by-passed oil zones in heterogeneous reservoirs can be identified by "low difference zones" in generally mobile reflector packets (Yilmaz, 2003). Usually, a simple difference of the 3D seismic volumes will not yield satisfactory results due to small-scale fluctuations in both arrival times and amplitudes, making time-shift analysis an important process to match the reflection events. These time-shift values have been shown to be a valuable source of information themselves (Hall et al., 2002a; Hatchell et al., 2005b), considering their sole dependence on wavefield kinematics, time shifts tend to be a more robust measurement than amplitude differences (Johnston, 2013b).

Considering normal incidence on a horizontal layer of thickness z and a P-wave velocity v with a traveltime t, we can express the changes in traveltime as:

$$\frac{\Delta t}{t} = \frac{\Delta z}{z} - \frac{\Delta v}{v}, \tag{2.1}$$

for homogeneous isotropic v and small changes in z and v. Originally developed in Hatchell et al. (2005b), with a rigorous integral derivation presented in MacBeth et al. (2019).

The vertical strain $\frac{\Delta z}{z}$ directly relates to the geomechanical strain ξ_{zz}, describing the vertical strain on the vertical surface of a infinitesemal element (Herwanger, 2015). Independently Hatchell et al. (2005b) and Røste et al. (2006) developed a single-parameter solution to relate velocity changes and vertical strain

$$\frac{\Delta v}{v} = -R\xi_{zz} \tag{2.2}$$

with R being the single parameter Hatchell-Bourne-Røste (HBR)-factor (Hatchell et al., 2005a; MacBeth et al., 2019). The HBR being a lithological constant, we can relate (2.2) and (2.1) and obtain a direct relationship between the vertical strain ξ_{zz} and the time shift Δt for a given lithology with property R

$$\Delta t = t \cdot (1 + R) \cdot \xi_{zz}. \tag{2.3}$$

Contingent on the assumption of zero-offset incidence, homogeneous velocity and isotropy, time shift extraction is mostly performed in z-direction by comparing traces directly. Prominently, the 1D windowed cross-correlation is used due to its computational speed and general lack of limiting underlying assumptions (Rickett et al., 2001). The main drawback of this method is, however, that the result is highly dependent on the window-size and susceptible to noise. Other methods for post-stack seismic time shift extraction include Dynamic Time Warping (DTW) (Hale, 2013a) and inversion-based approaches (Rickett et al., 2007).

More recently research into pre-stack time shift extraction and 3D-based methods is conducted. These methods relax the constraints of some assumptions of 1D applications (Ghaderi et al., 2005; Hall et al., 2002b). 3D time shifts have the ability to capture subsurface movement of reflectors and account for 3D effects of the $\Delta R/\Delta S$ acquisition mismatch, which effect seismic illumination.

Qantitative Interpretation (QI) extends the interpretation of 4D changes to estimate fluid saturation and pressure changes within the reservoir. The subsurface changes recorded by the seismic data can be related numerically to subsurface changes. The process of extracting causal information from imaging data is called inversion. The underlying phenomena interact with several possible and physical explanations for the same seismic response, which makes the inversion process non-unique and often reliant on prior information. The decoupling of pressure and saturation changes is non-trivial and relies on pre-stack or angle-stack information (Landrø, 2001). This process is, however, highly desirable with the benefit of quantifying the subsurface changes from seismic data directly.

Active areas of research in 4D seismic are the use of 4D seismic data to estimate saturation and pressure changes quantitatively particularly in volumetric applications as opposed to map-based approaches. However, these approaches often depend on reliable rock-physics models, an area of research in model-based approaches. Moreover, there's active research in moving to volumetric approaches in time-shift estimation and quantitative pre-stack analysis. Additional research in extractive data-based methods and model-based approaches investigate how much information is available directly from the data and what information is available from the modelling feedback-loop.

2.2 Machine Learning

Machine Learning (ML) is the discipline of defining a statistical or mathematical models based on data. These ML models are either trained in a supervised or unsupervised fashion, which usually results in them learning a decision boundary, or a representation or structure of the data respectively. Historically, ML has been an interest in geoscience but has not gained momentum due to sparse data, computational capability, and availability of algorithms. Geoscience data was often not available and still is often not available

with a reliable ground truth. However, particularly NNs have found broad interest in geophysical applications, Bayesian methods are often used in inversion schemes and recent software developments have changed the research entirely.

Recently, the subfield Deep Learning (DL) has reignited interest in the wider field of ML by outperforming rule-based algorithms on computer vision tasks, such as image classification and segmentation (Bishop, 2016). These developments have propelled developments in other non-related fields such as biology (Ching et al., 2018), chemistry (Schütt et al., 2017), medicine (Shen et al., 2017) and pharmacology (Kadurin et al., 2017). DL utilizes many-layered artificial NN to approximate an objective function. In recent years the open source movement, democratization of access to computing power and developments in the field of DL have rekindled interest in applications of ML to geoscience. The availability of free open source libraries such as skikit-learn (Pedregosa et al., 2011) has made ML methods and several tools for the application of rigorous statistical evaluation of experiments without explicit expert knowledge widely available. Furthermore, Tensorflow (Martín Abadi et al., 2015), PyTorch (Paszke et al., 2017), and Keras (Chollet et al., 2015) have made NNs easily accessible and provide experimentation capabilities to transfer recent developments in ML research to other scientific fields.

Algorithms and methods in ML can be organized in different ways. Two ways to categorize algorithms are based on the training or based on the learned distribution. In training, these algorithms can be categorized into supervised and unsupervised methods, where supervised methods learn the functional mapping from x, being the data, to y, being the ground truth or label for the data. When the ground truth is not known, unsupervised methods can be applied to determine structures and relationships within the data. Semi-supervised, and weakly supervised try to propagate partial labels to similarly distributed data and then learn the supervised mapping f(x) = y. Alternatively, ML algorithms can be categorized into generative methods that learn the joint probability distribution or discriminative methods that learn a decision boundary to optimally separate data. Additionally, methods can be distinguished by application. Assigning labels to data is called classification. The general, continuous application to map data from the input to the output domain is called regression. Finding relationships and agglomerations of data is called clustering. Most algorithms can be applied to several of these categories, such as support vector machines that can function as classifier and regressor.

Applications in ML are quickly evolving and many are improved by mathematical insights, engineering features and increased availability of data. This thesis focuses on the application of NNs, which come in different implementation details and particularly NN architectures are often re-implemented with slight differences that deviate from the original published architecture. Particularly in NN we have to focus on the most practical building blocks, to be able to give a comprehensive overview.

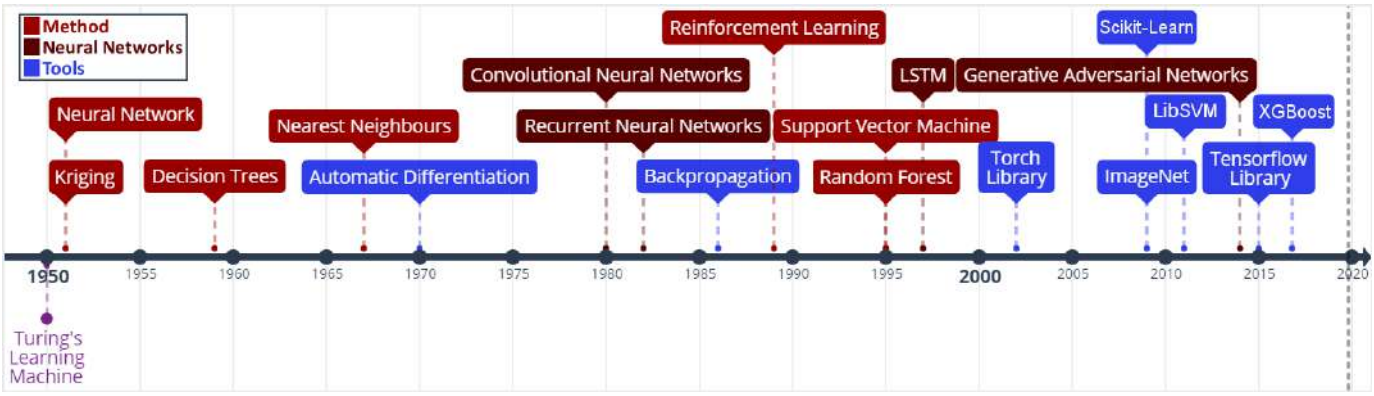


Figure 2.1: Selection of notable milestones in machine learning.

2.2.1 History of Machine Learning

Creativity, learning, and intelligence with regard to computers have been discussed as early as of the first programmer Ada Lovelace (Taylor, 1843).

> "The Anlytical Engine has no pretensions whatever to *originate* any thing. It can do whatever we *know how to order it* to perform. It can *follow* analysis; but it has no power of *anticipating* any analytical relations or truths. Its province is to assist us in making *available* what we are already acquainted with. This it is calculated too effect primarily and chiefly of course, through its executive faculties; but it is likely to exert an *indirect* and reciprocal influence on science itself in another manner." – Note G, Page 689, Ada A. Lovelace. (Taylor, 1843); Emphasis taken from source text.

This notion was challenged by Alan Turing (Turing, 1950) who proposed the "Learning Machine", which specifically predict genetic algorithms, a metaheuristic that finds application in optimization and search problems. Evolutionary computing and genetic algorithms specifically can perform some machine learning tasks (Goldberg et al., 1988). This is generally considered the commencement of Artificial Intelligence (AI) and ML, however, they rely heavily on earlier developments in statistics such as the Bayesian theorem (Bayes, 1763) and Markov processes (Markov, 1906; Markov, 1971). The first method, we include on the timeline in Figure 2.1 is "kriging" (Krige, 1951), which is based on Gaussian Processes, these form an important category of non-parametric machine learning these days. Gaussian processes are often also attributed to work of Kolmogorov (1939) on time series. Another method was developed to mimic the human brain, namely Neural Networks (NNs). The construction of the first NN machine by Minsky (Russell et al., 2010) was soon followed by the "Perceptron", a binary decision boundary learner (Rosenblatt, 1958). The decision is made according to

$$
\begin{aligned}
o_j &= \sigma\left(\textstyle\sum_i w_{ij} x_i + b\right) \\
&= \begin{cases} 1 & \textstyle\sum_i w_{ij} x_i + b > 0 \\ 0 & \text{otherwise} \end{cases}
\end{aligned} \tag{2.4}
$$

which describes a linear system of the input data x, the weights w and bias b and a binary activation funtion σ. The linear system is still used in modern neurons, however, the activation σ is usually a Rectifier function. Shortly after, Belson (1959) describe the first Decision Tree (DT), which learns hierarchical decision systems. The next method, k-Nearest Neighbour (KNN) search, was introduced by Cover et al. (1967) to solve the traveling salesman problem. Two decades later Q-learning (Watkins, 1989) introduces a method to reinforcement learning that is still used to this day. The final two methods in the timeline were introduced in 1995. Random Forests (RFs) (Ho, 1995) introduce ensemble learning of weak learning Decision Trees (DTs). Support Vector Machine (SVM) (Cortes et al., 1995) introduce a strong learner that aims to maximize the margin between classes.

These methods have been improved upon over the decades. Specific milestones that accelerated further developments in NN are automatic differentiation (Linnainmaa, 1970) and consequently applying this to backpropagate errors in Deep Neural Networks (DNNs) (Rumelhart et al., 1988). Backpropagation itself as a concept existed earlier (Kelley, 1960; Bryson, 1961), followed by a simplification by using the chain rule (Dreyfus, 1962). These enable effective implementation of NNs today. Moreover, open sourcing the Torch library (Collobert et al., 2002) made and assembling the ImageNet database (Deng et al., 2009) has accelerated developments in computer vision and enabled modern developments in deep learning. In the same year of 2009 the library Scikit-Learn (Pedregosa et al., 2011) was established, which introduced a common open source Application Programming Interface (API) (Buitinck et al., 2013) for a diverse and growing set of shallow machine learning models (e.g. SVMs, RFs, KNNs, shallow NNs). Scikit-learn has had a profound impact on machine learning applications across the sciences and the API is modelled in other open source libraries. Chang et al. (2011) introduced a widely used implementation for Support Vector Machine (SVM), which is also used in Scikit-Learn. Recently, the Tensorflow library (Martín Abadi et al., 2015) was introduced for open source deep learning models, with some different design choices than Pytorch. In this open environment fueled by competitions (e.g. ImageNet (Russakovsky et al., 2013), Netflix Prize (Bennett et al., 2007), Kaggle (Goodfellow et al., 2013)) XGBoost (Chen et al., 2016), a library for extreme gradient tree boosting was developed.

Recent developments in deep learning are based in Neural Networks (NNs), hence, we highlight some key developments in Figure 2.1. Convolutional Neural Networks (CNNs) are essential in the modern computational vision systems, they were inspired by the concept of Neocognitron (Fukushima, 1980; LeCun et al., 2015). In the same decade Recurrent Neural Networks (RNNs) were introduced implemented as Hopfield Networks (Hopfield, 1982). While Hopfield networks are not a general RNN, they provide content-adressable memory with the internal state memory. Hochreiter et al. (1997) implement

the Long Short-Term Memory (LSTM), which contain internal states (i.e. memory) that can process temporal sequences, still used and performing to the state-of-the-art in sequence analysis and Natural Language Processing (NLP) to this day. Recently, Generative Adversarial Network (GAN) (Goodfellow et al., 2014c) introduced a system of NNs that can create new samples from a distribution. The GAN consists of two NNs, a generator and a discriminator, which generate samples from a noise distribution and judge the validity of the sample respectively. We discuss NNs in more detail in Section 2.2.2

2.2.2 Neural Networks (NNs)

Neural Network (NN) as a class of ML algorithms are very diverse and versatile. NNs have persisted for decades and their nomenclature has changed in this time. NNs were long called Artificial Neural Network (ANN), which has changed to simply NN, usually prepended with the class of Neural Network, namely Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), Deep Neural Network (DNN), which I will discuss in more detail.

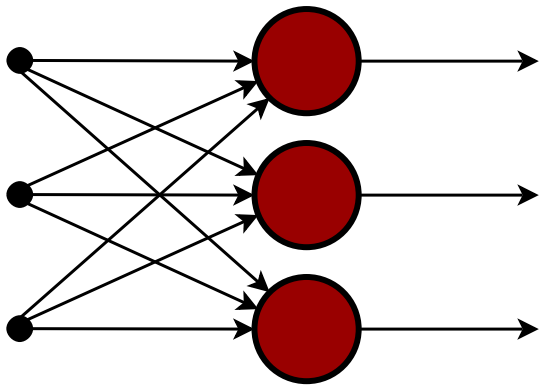
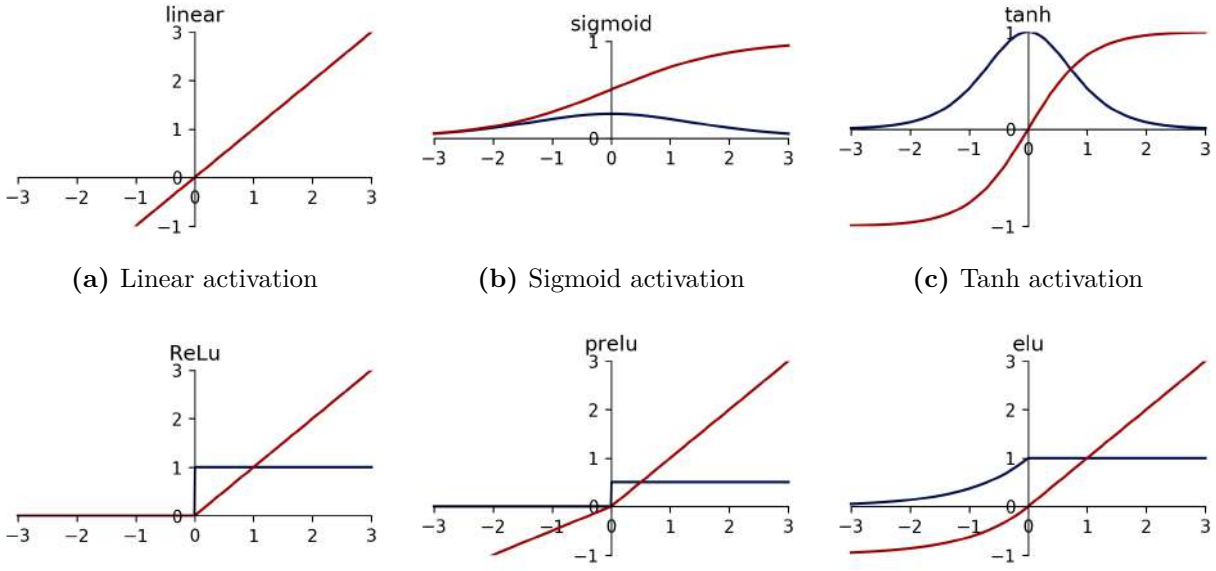


Figure 2.2: Basic NN with three inputs that are densely connected to three output neurons by weights.

Neural Networks (NNs) can be approached from several theoretical bases. Mathematically, NNs are directed acyclical graphs with edges and nodes. In neural computation, these are generally referred to as weights and nodes or neurons. In Figure 2.2, we present a simple densely connected Multi-Layer Perceptron (MLP) with three inputs and three outputs. This configuration is equivalent to a linear regression model. The inputs are distributed across the nodes, and each weight is multiplied with a weight inherent to that graph edge. During the training of this machine learning model, these weights get adjusted to obtain a generalizable result. Each node sums the contributions of these weights and possibly a bias, which is trainable but does not take input data. This amounts to each node performing

$$a_j = \sigma\left(\sum_i w_{ij} x_i + b\right), \tag{2.5}$$

with a signifying the activation at a node, i, j being the index of the source and target node respectively, w being the trainable weight, and b being the trainable bias, and σ representing an activation function. Activation functions are an active topic of research, but the generally perform a non-linear transformation of the activation at the node.



(a) Linear activation **(b)** Sigmoid activation **(c)** Tanh activation

(d) ReLU activation **(e)** PReLU activation ($\alpha = .5$) **(f)** ELU activation ($\alpha = 1$)

Figure 2.3: Common Activation functions (red) and derivatives (blue). The linear activation does not modify the data. The sigmoid and tanh functions are mainly used to limit output activations to a range of values. The ReLU, PReLU and ELU activations are different iterations of rectifiers that are used in Deep Neural Networks.

In Figure 2.3 I present common activation functions used in NNs. The activation functions introduce non-linearities into the network to transform the linearly scaled input to arbitrary non-linear outputs. The mathematical function in Figure 2.3(b) and Figure 2.3(c) are used less, because of the vanishing gradient problem (Hochreiter, 1991). These occur in the extrema of both functions, where the function saturates and the gradient is close to zero for large values of x. Rectifiers presented in Figures 2.3(d) to 2.3(f) circumvent this problem by one-sided saturation.

Training the Model Before training, each weight and bias is assigned an initial number that is drawn from a distribution appropriate to the network architecture and data (LeCun et al., 2012; Glorot et al., 2010; He et al., 2015). These strategies collectively initialize weights in a pseudo-random way within limits. The data is then passed through

the network, which calculates a result. This result is then compared to the ground truth, using a loss function (e.g. Mean Absolute Error (MAE), Mean Squared Error (MSE)). The resulting error Δt is then used to correct the weights and biases in the network, calculating the correction per weight Δw_{ij} recursively (for many-layered networks).

$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = -\eta \delta_j a_i, \tag{2.6}$$

with η being the learning rate and δ being

$$\delta_j = \begin{cases} \sigma'(\text{net}_j)\Delta t & \text{if } j \text{ is output node,} \\ \sigma'(\text{net}_j) \sum_{j-1} \delta_{j-1} w_{j(j-1)} & \text{if } j \text{ is hidden node.} \end{cases} \tag{2.7}$$

Therefore, hidden nodes are reliant on the result δ_{j-1} of the node at index $j-1$ (Goodfellow et al., 2016). The training of the model can be done on a per-sample basis, which is Stochastic Gradient Descent (SGD) or in the case of noisy inputs, the mean error of several samples can be calculated to perform mini-batch gradient descent. Iteration over forward and backward passes adjusts the weights to predict the correct result.

Modern deep NNs are trained on Graphical Processing Units (GPUs) that are optimized for matrix multiplications instead of Central Processing Units (CPUs) that are magnitudes slower. However, more recently task-specific hardware such as Field Programmable Gate Arrays (FGPAs) and Tensor Processing Units (TPUs), which work closely with the Tensorflow (TF) library are being developed and made available in cloud infrastructures.

The optimization of the backpropagation is performed using SGD or other gradient-based optimizers such as the Adam optimizer (Kingma et al., 2014). However, during training of the NN, it is important to ensure that the network learns a general relationship instead of memorizing the input data. This memorization is called overtraining, or overfitting. Overfitting can be avoided by regularizations like weight decay (Krogh et al., 1992) and Nesterov momentum (Sutskever et al., 2013), which modify the optimization loop. Alternatively, methods like Dropout (Hinton et al., 2012) and Batch Normalization (BN) (Ioffe et al., 2015) modify the network at training time. Moreover, a diverse training set and train-val-test split help avoid overfitting and ensure generalization of the trained model.

The train-val-test split separates the data into three parts. The training and validation set are available during training and hyperparameter tuning, the test set, however, should only be used once to ensure generalization of the model. The train test is used during the optimization loop, the actual training of the model, with the validation set ensuring generalization of the model to unseen data within the loop. In and of itself, the train and validation data would be sufficient, if no other changes to the model were made based on the results of the validation data. Since hyperparameter tuning and model selection are a common procedure today, these present an implicit source of information leakage from the validation set into the data. The hyperparameter tuning will often pose an optimization loop in itself that optimizes based on the results on the "unseen" validation set, essentially implicitly fitting the model to the validation data, therefore, a separate test set is necessary to ensure true generalization.

2.2.2.1 Feed Forward Networks

Feed forward Neural Networks (NNs) or MLPs are the simplest for of NN. In its simplest form it uses a set of linear equations to approximate a function. The network can be described as a graph with edges and nodes. In the neural information community the nodes are often named neurons. These neurons are arranged into layers in Figure 2.4. The first layer in a NN is the input layer with a number of nodes corresponding to the number of input data points. The input nodes are connected to the next layer by the graph's edge. The next node can be the output layer. The weights between subsequent are floating point numbers that scale each input point and determine the value at the output nodes.

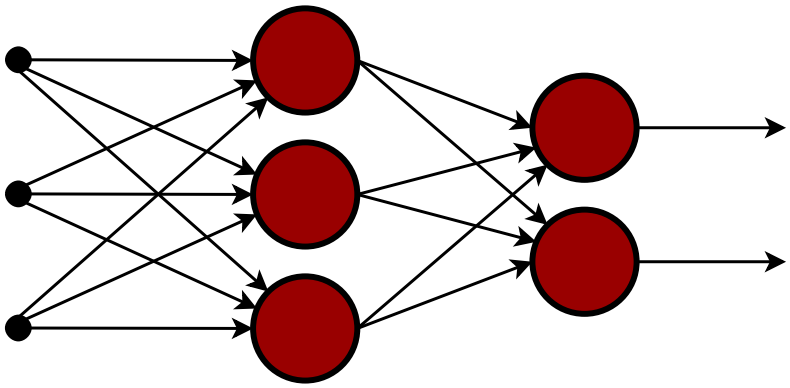


Figure 2.4: Feed forward NN with three input neurons that are connected to a single hidden layer with three neurons. The hidden layer is densely connected to two output neurons.

NNs gain their powerful learning capabilities from adding layers (see Figure 2.4) in between the input and output node and applying a non-linear activation function. Non-linear activations scale the input from the edge at each neuron. Historically, these have been straight-forward mathematical functions such as $tanh()$ and $sig()$ (cf. Figure 2.3). These suffer from some short-comings that were overcome to leverage multi-layered Deep Neural Networks (DNNs).

2.2.2.2 Deep Neural Networks (DNNs)

Improvements in computational power made it possible to train many-layered NNs (see Figure 2.5). These Deep Neural Networks (DNNs) are at the core of recent developments in Deep Learning (DL), leading to the re-implementation of many algorithms into openly available libraries, which has led to further innovative uses of these building blocks. These networks leverage the combinatorial power of NN layers. In deep NNs gradient propagation led to exploding or vanishing gradients before. New non-saturating activation functions lead to stabilization of training DNN (cf. Figure 2.3).

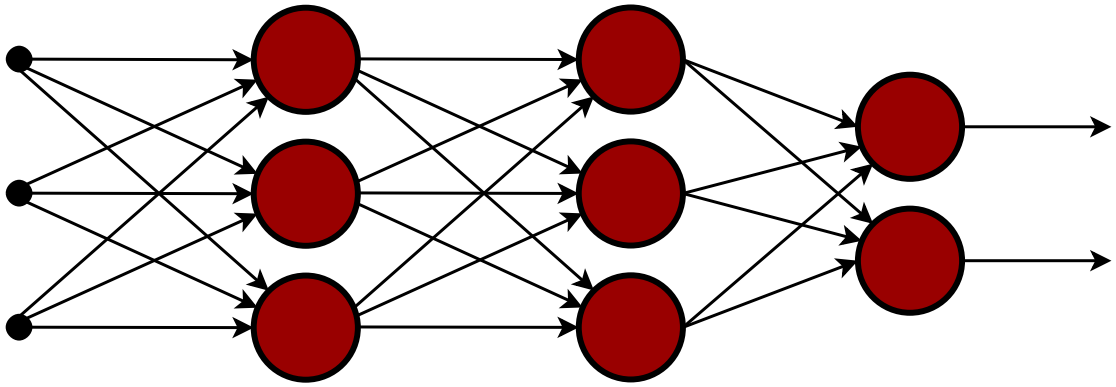


Figure 2.5: Deep Feed forward NN with two hidden layers with three neurons each, densely connected to three inputs and two output neurons. Deep networks are NNs that contain more than one hidden layer.

2.2.2.3 Self-Organizing Maps (SOMs)

Self-Organizing Map (SOM), also named Kohonen-networks (Kohonen, 1982) are a special case of networks that do not modify the flow of data from the input to the output nodes. They treat each data point as a node and adjust the weights between each node in on a similarity metric. These tend to perform well on spatially correlated data and find good adoption in geoscience.

2.2.2.4 Recurrent Networks

A special configuration of NN is the Recurrent Neural Network (RNN). These networks use edges that feed back into the network. RNNs are used in two applications in ML. They can preserve hidden states, which gives them temporal context sensitivity. Application two is time series analysis similar to feed-forward NNs, where the input is a time step that can be analyzed within the context of surrounding time steps. These RNN represent cyclic directed graphs of computation, as opposed to the other types of NN we discuss, which are acyclic directed graphs. In Figure 2.6 we show the changes of a simple RNN graph compared to a feed forward NN in Figure 2.4. The RNN loops back into itself, which is often regarded as the internal state or feedback. This internal state enables content addressable memory and good performance on sequential data such as time series and language.

Hopfield Networks are one type of recurrent networks that model the human memory. Hopfield networks and their subclasses can be used for pattern recognition. They are guaranteed to find a pattern, however, they are known to converge to local minima. Boltzman machines are configured like Hopfield networks, in contrast to deterministic Hopfield networks, their response to an input is stochastic. Boltzman machines draw from a joint distribution, making them a generative model.

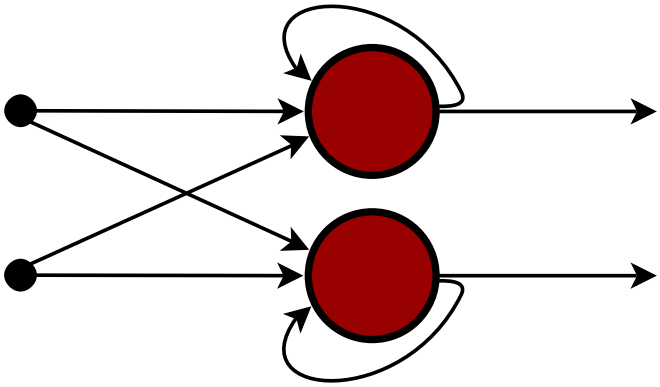


Figure 2.6: Recurrent NN that connects two input neurons to two recurrent neurons. These recurrent neurons feed back into themselves, which signifies the state of the neuron. RNN neurons are more complicated internally than the neurons in CNNs accomodating the state memory.

Long Short-Term Memory (LSTM) is a type of RNN that models memory. Details differ in implementations of Long Short-Term Memory (LSTM), however the main criteria are three gates and an inner cell.

- Input Gate
- Forget Gate
- Output Gate

The input gate regulates the contribution of input values to the internal cell. The forget gate regulates the persistence of values in the cell. Finally, the output gate regulates the contribution of the input value to the output value convolved with the cell state.

2.2.2.5 Convolutional Networks

Convolutional Neural Network (CNN) were developed in computer vision to automatically learn a filter that spatially correlates data. The convolutional kernels are computationally efficient due to weight sharing, making them feasible for very deep networks (cf. Section 2.2.2.2). CNNs have had the biggest influence on the renaissance of modern ML. These building blocks for NNs are very good for image data and data where spatially correlated information provides valuable context. It has therefore quickly gained attention in seismic interpretation and seismic data analysis. CNNs like other NNs are optimized by SGD, optimizing a defined loss over the chosen task.

For a two-dimensional CNN, the convolution of the $m \times n$-dimensional image G with a filter matrix f can be expressed as:

$$G^*(x,y) = \sum_{i=1}^{n} \sum_{j=1}^{m} f(i,j) \cdot G(x-i+c,\ y-j+c), \tag{2.8}$$

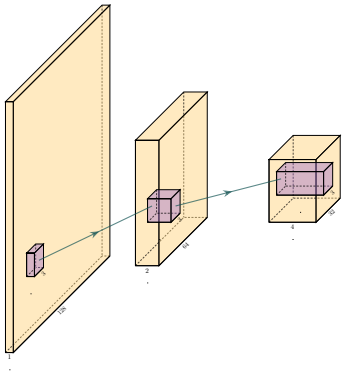


Figure 2.7: Schematic of a CNN filter (purple) in the image data (orange) in 2D. The filter passes over the image, extracting a filtered representation of the input image. The image is downsampled spatially by striding or pooling. Convolutional filters are efficient due to weight sharing.

resulting in the central result G^* around the coordinate c. Realistically, the calculation is done in the Fourier domain, due to the Convolution theorem reducing the computational complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$ with

$$\mathcal{F}\{f * g\} = k \cdot \mathcal{F}\{f\} \cdot \mathcal{F}\{g\}, \tag{2.9}$$

with $\mathcal{F}\{f\}$ denoting the Fourier transform of f and k being a normalization constant. This reduces the convolution to a simple multiplication in the Fourier domain, sped up by Fast Fourier transform (FFT).

Figure 2.7 shows the schematic of connected convolutional layers in a CNN. The network learns a specified number of 3×3 filters from the initial image. Strided convolutions with a step-size larger than 1 or Pooling layers are used to reduce the spatial extent of the image. The repeated downsampling of the image and extraction of convolutional filters has been shown to work for computer vision tasks. Historically, the CNN architecture AlexNet (Krizhevsky et al., 2012) was the first CNN to enter the ImageNet challenge and improved the classification error rate from 25.8 % to 16.4 % (top-5 accuracy). This has propelled research in CNNs, resulting in error rates on ImageNet of 2.25 % on top-5 accuracy in 2017 (Russakovsky et al., 2015).

2.2.2.6 Generative Adversarial Networks

Goodfellow et al. (2014b) introduced Generative Adversarial Network (GAN) as a combination of two CNNs. These Deep Convolutional Generative Adversarial Networks (DCGANs) exist in different modifications that draw from the original GAN, these modifications add more regularization and other feedback loops, as GANs are notoriously difficult to train without careful fine-tuning. These modifications include Wasserstein losses (Arjovsky et al., 2017), and gradient penalization (Gulrajani et al., 2017) for regularization, or cycle-consistent loss for unsupervised training (Zhu et al., 2017).

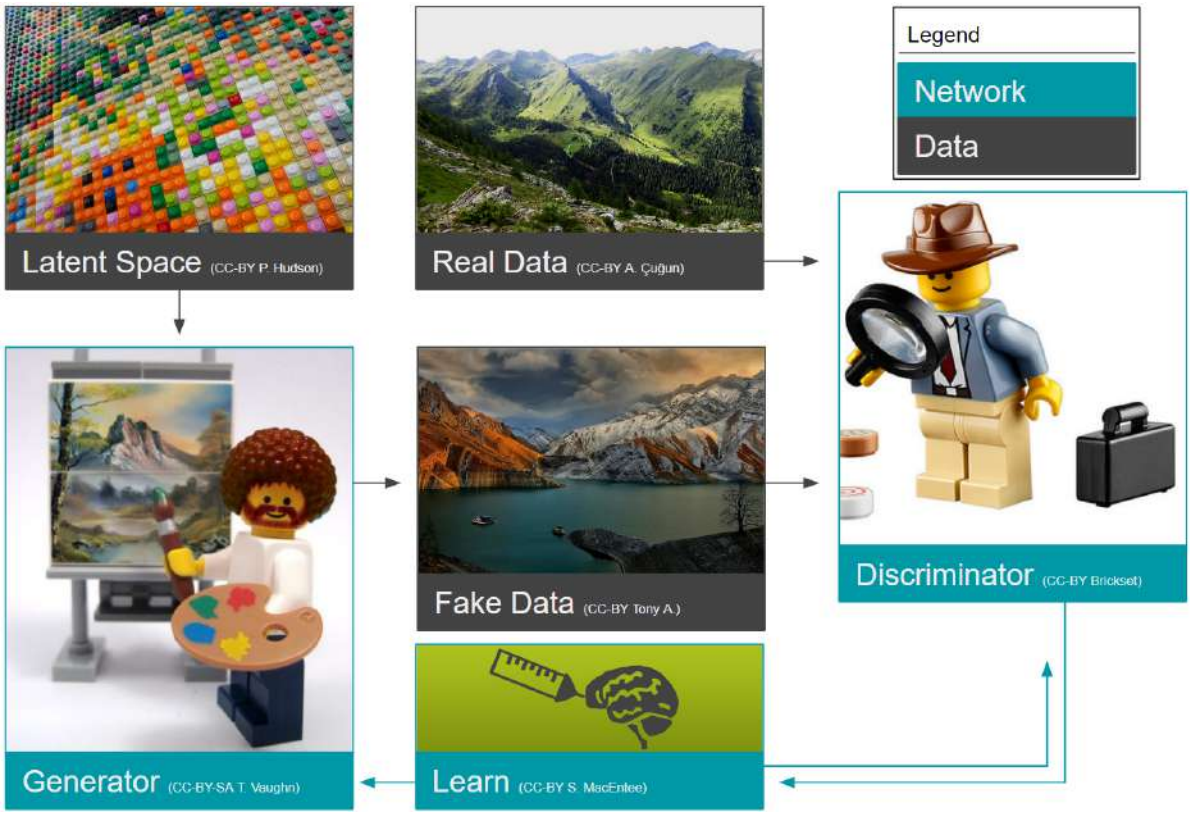


Figure 2.8: Schematic of a Generative Adversarial Network. The generator samples a latent space to generate fake data. The discriminator randomly obtains real or fake data and decides whether it was created by the generator or a real sample. The networks learn by gradient descent gaining information regardless of the discriminator being right.

Figure 2.8 shows the basic working of GANs. The arrows are colored in blue and grey, where the blue paths show network feedback and grey shows the progression of data. These networks learn from each other, where the generator draws from latent space (a noise vector) to create a fake version of a target. The discriminator tries to discern whether the presented data is real or generated from the adversarial generator. These networks leverage game theory to outperform each other and comparative networks. They reach a Nash equilibrium during training, which describes the concept on a non-cooperative game reaching steady state (Nash, 1951).

2.2.3 Neural Architectures

Neural Networks can generally be assembled in different architectures. In Figure 2.10 we present reported performances of neural architectures on the classification task of the ImageNet challenge. The colors in this figure express different classes of architectures. Early networks that broke ground as the new state-of-the-arts in image classification are

the AlexNet, VGG-16, and VGG-19. These networks clearly do not leverage some tricks that modern CNNs implement, the VGG-16 with a relatively high amount of parameters is known to generalize well on transfer learning tasks however (Dramsch et al., 2018c).

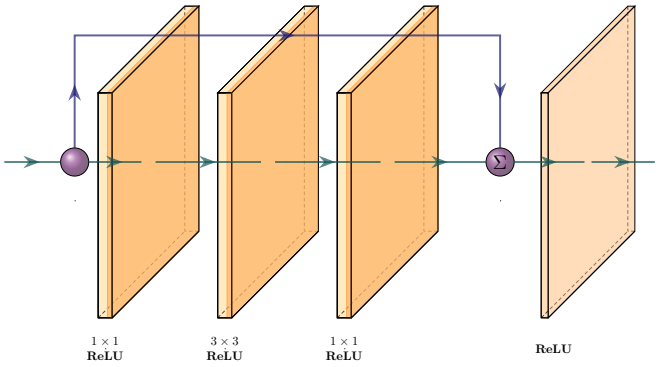


Figure 2.9: Resnet Block with two 1×1 convolutional layers that frame a 3×3 convolutional layer with ReLU activation each. The result being added with the original data, also known as identity..

Research into deep convolutional networks showed that the data in the network would lose signal with increasing depth. Hence, the limitation of VGG at 19 layers. Residual blocks introduced a solution to this problem by implementing a shortcut between the original data and the output from the block. Figure 2.9 presents the original ResNet block architecture, which was used in ResNet-50 and ResNet-101 in Figure 2.10 (He et al., 2016). Details on ResNet blocks differ, the main take-away being the sum or concatenation of the original data with the block output. DenseNets (Huang et al., 2017a) and Inception-style networks (Szegedy et al., 2015) are other approaches to build deeper NNs.

The categories of AmoebaNet, NASNet, and EfficientNet are a more recent development in neural architecture research, based on Neural Architecture Search (NAS). The AmoebaNet is based on Evolutionary Computing and hand-tuning the solution to search for an ideal neural architecture to solve the task (Real et al., 2019). The NASNet fixes the overall architecture, but uses a controler RNN to modify the blocks within the architecture (Zoph et al., 2018). The EfficientNet architecture was also acquired by NAS, by optimizing for both accuracy and FLOPS to reduce the computational cost (Tan et al., 2019b). Moreover, Tan et al. (2019b) derives a method of compound scaling for deep neural networks. While ResNet-50 and ResNet-101 differ only in depth, the authors derive a relationship between depth, width and resolution-scaling of deep neural networks.

Apart from building deeper networks for image classification, the neural architectures can serve as a forcing function to the task the network is built for. Encoder-Decoder networks will compress the data with a combination of downsampling layers, which in the case of a computer vision could either be strided convolutions or pooling layers after convolutional layers. During these operations, the number of filters increases, while the

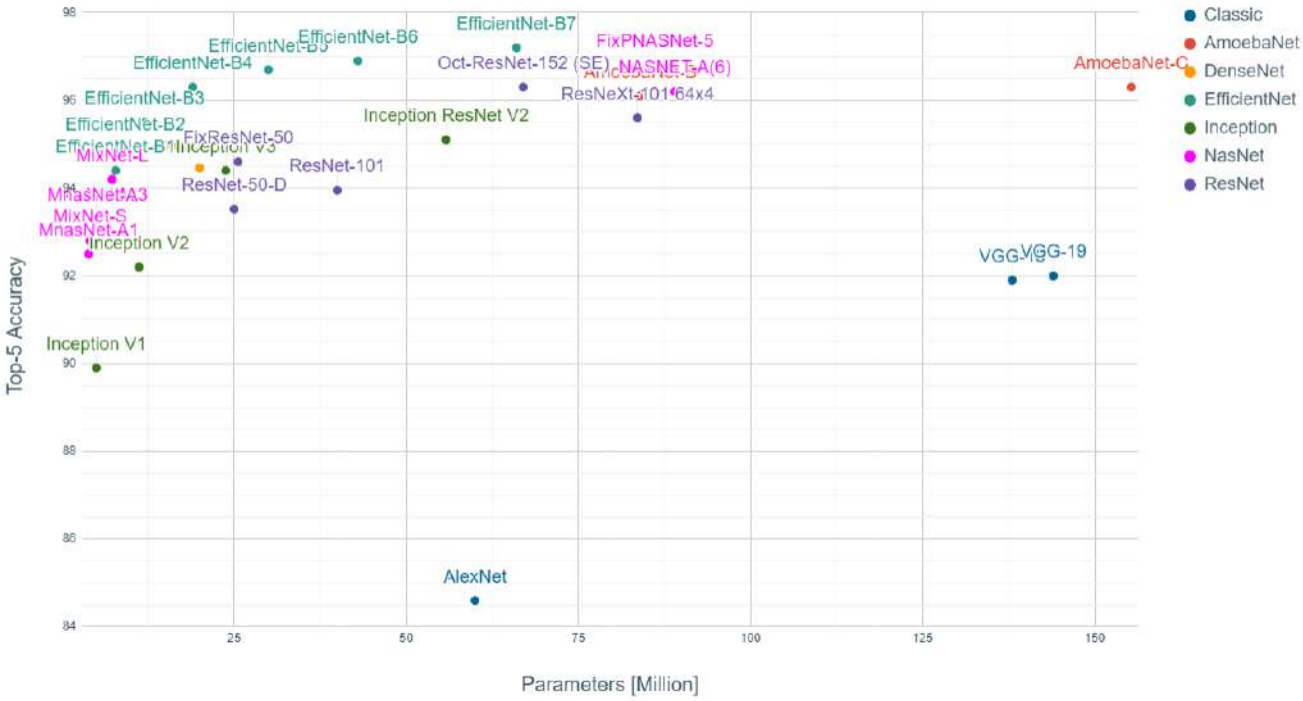


Figure 2.10: Top-5 Accuracies of Neural Architectures on ImageNet plotted against Million Parameters, color-coded to similar network type. Data and references shown in Table A.1.

spatial extent is diminished significantly. This encoding operation is equivalent to a lossy compression, with the low-dimensional layer called ”code” or ”bottleneck”. The bottleneck is then upsampled by either strided transpose Convolutions or upsampling layers that perform a specified interpolation. This is the Decoder of the Encoder-Decoder pair. These networks can be used for data compresssion in AutoEncoders (AEs), where the decoder restores the original data as good as possible (Hinton et al., 2006). Alternatively, the Decoder can learn a dense classification task like semantic segmentation or seismic interpretation.

U-Nets present a special type of encoder-decoder networks, that learn semantic segmentation on from small datasets (Ronneberger et al., 2015). They form a special kind of Fully Convolutional Network (FCN) shown in Figure 2.11. Originally developed on biomedical images, the network found wide acceptance in label sparse disciplines. The Unet implements shortcut connections between convolutional layers of equal extent in the Encoder and Decoder networks. This alleviates the pressure of the network learning and reconstructing the output data from the bottleneck in isolation.

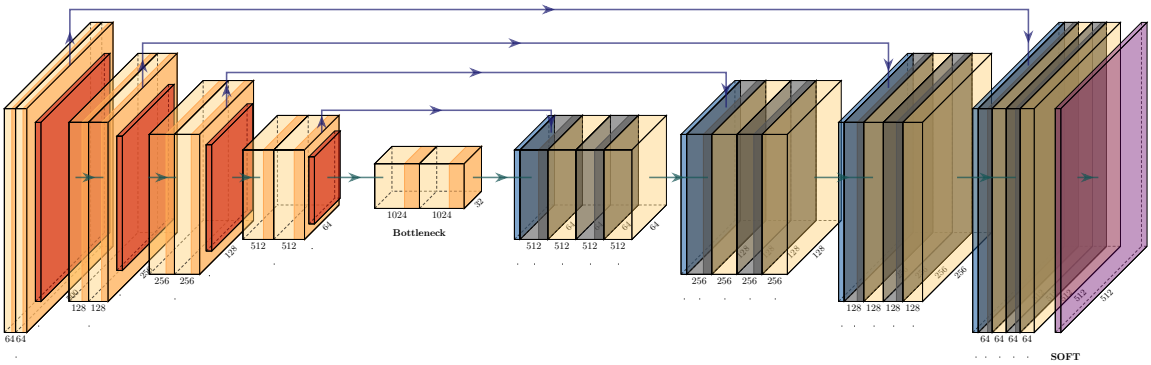


Figure 2.11: Unet after Ronneberger et al. (2015) using 2D convolutional layers (yellow) with ReLU activation (orange) and skip connection between equal-dimensional layers. The Encoder uses pooling (red), while the Decoder uses Upsampling layers (blue), witha final SoftMax layer (purple) for classification / semantic segmentation.

2.3 Machine Learning in Geoscience

The development of the subfield of deep learning has lead to advances in many scientific fields that are not directly related to the larger field of artificial intelligence. This section focuses on historic use-cases of machine learning models in geoscience and evaluate these in the context of recent advances in deep learning. I provide an overview of supervised and unsupervised methods that have persevered. Furthermore, I distinguish implementations of deep neural network topologies and advanced machine learning methods in geoscientific applications. I go on to investigate where these methods differ from previously unsuccessful attempts at application.

Early on Machine Learning (ML) has been reviewed in a geophysical context. Early publications of ML in geoscience apply NNs to geophysical problems. Particularly seismic processing lends itself to explore NNs as general functional approximator (Hornik et al., 1989). McCormack (1991) review of the emerging tool of neural networks in 1991. He highlights the application of pattern recognition and is very succinct in describing basic math associated with neural computing. The wording of most parts has changed, as compared to today. Generally this gives a good baseline and McCormack gives a good illustration and overview with examples in well log classification and trace editing. The author summarizes NN applications over the 30 year prior to the review and hightlights automated well-log analysis and seismic trace editing. The review comes to a conclusion that these methods show promise as general approximators.

Baan et al. (2000) review the most recent advancements in Neural Networks (NNs) in geophysical applications. It goes into much detail on the neural networks employed in 2000 and the difficulties in building these models and training them. They identify the following subsurface geoscience applications through history: First-break picking, electromagnetics, magnetotellurics, seismic inversion, shear-wave splitting, well log anal-

ysis, trace editing, seismic deconvolution, and event classification. The authors evaluate the application of NNs as subpar to physics-based approaches. The paper concludes that neural networks are too expensive and complex to be of real value in geoscience. Generally, this review focuses very much on exploration geoscienc.

Mjolsness et al. (2001) review ML in a broader context outside of exploration geoscience. They illustrate recent successes of ML in analyzing sattelite data and computer robotic geology. The authors include graphical models, Random Markov Models (RMMs), Hidden Markov Models (HMMs), and SVMs. They further highlight limitations to vector data, therefore failing richer data such as graphs and text data. Moreover, the authors from NASA JPL go into detail on pattern recognition in automated rovers to identify geological prospects on Mars. They state:

> "The scientific need for geological feature catalogs has led to multiyear human surveys of Mars orbital imagery yielding tens of thousands of cataloged, characterized features including impact craters, faults, and ridges." - (Mjolsness et al., 2001)

The authors evaluate how especially the introduction of SVM have allowed the identification of geomorphological features without modeling the processes behind. Further they mention recurrent neural networks in gene expression data, a method that has experienced a renaissance in deep learning.

2.3.1 History of Machine Learning in Geoscience

Machine learning, statistical, and mathematical models have a long history in geoscience. Markov models have been used to describe sedimentology as early as the 1970s (Schwarzacher, 1972) and the use of k-means in geoscience as early as 1964 (Preston et al., 1964). In geophysics applications of NNs to perform seismic devonvolution were published in the 1980s Zhao et al. (1988). Early tree-based methods were chiefly used in economic geology and exploration geophysics for prospectivity mapping with Decision Trees (DTs) (Newendorp, 1976; Reddy et al., 1991). SVM has early on been applied to AVO classification Li et al., 2004 and geological facies delineation for hydrological analysis (Tartakovsky, 2004). This thesis mostly focuses on the application of NNs, however, we give an additional overview of geoscientific applications of shallow ML.

2.3.1.1 Machine Learning Applications in Geoscience

Early applications of neural networks were prominent in seismic data processing and analysis. Zhao et al. (1988) use a NN to perform seismic deconvolution early on. An application of seismic inversion with NNs was published by Röth et al. (1994). Early ML-based electromagnetic geophysics performs subsurface localization (Poulton et al., 1992) and magnetotelluric inversion via Hopfield NNs (Zhang et al., 1997). Feng et al. (1998) applied NN to model geomechanical microfractures in triaxial compression tests. Interestingly, Legget et al. (1996) used a combination of Self-Organizing Map (SOM)

and back-propagation NNs that function similar to modern day Convolutional Neural Networks (CNNs) to perform 3D horizon tracking (Leggett et al., 2003). With the recent DL explosion, papers on Automatic Seismic Interpretation (ASI) have gotten very popular, given the similarity to 2D segmentation tasks (cf. Table B.1).

Modern CNNs have been applied to a wide variety of geoscience problems including seismic inversion (Araya-Polo et al., 2018), and applications in seismology such as first break picking (Ross et al., 2018a) or event classification (Zhu et al., 2018; Ross et al., 2018b). In 2017 the application of Generative Adversarial Networks in geoscience in digital rock modelling (Mosser et al., 2017), geostatistical modelling (Laloy et al., 2017) and seismic inversion (Mosser et al., 2018d; Mosser et al., 2018c). Further applications extend to geochemical anomaly detection (Zuo et al., 2018) using Variational AutoEncoders (VAEs) and hydrogeological modelling (Sahoo et al., 2017). Common applications include Ground Penetrating Radar (GPR), various applications in seismic processing, analysis and interpretation, as well as seismology, listed in detail in Table B.1.

Recently, some applications of Deep Neural Networks to predict earthquake aftershocks (DeVries et al., 2018) has been called into question by the publication "One neuron versus deep learning in aftershock prediction" (Mignan et al., 2019b). Criticizing the original publication for over-engineering a problem that is well-defined on less input data. A common critique of ML and big data analytics by classical statistics and rigorous data science (Mignan et al., 2019a).

Support Vector Machines have early-on been used for seismic data analysis (Li et al., 2004) and the popular approach of Automatic Seismic Interpretation (Liu et al., 2015; Di et al., 2017b; Mardan et al., 2017). Additionally, early applications include seismological volcanic tremor classification (Masotti et al., 2006; Masotti et al., 2008) and Ground Penetrating Radar analysis (Pasolli et al., 2009; Xie et al., 2013). The 2016 SEG ML challenge was introduced using a SVM baseline (Hall, 2016), with several other investigations into SVMs for well log analysis (Anifowose et al., 2017a; Caté et al., 2018; Gupta et al., 2018; Saporetti et al., 2018). Moreover, this method has been applied to seismology for event classification (Malfante et al., 2018) and magnitude determination (Ochoa et al., 2018). Considering the strong mathematical foundation of SVMs, they have been applied to applied to a variety of geoscience problems such as microseismic event classification (Zhao et al., 2017b), seismic well ties (Chaki et al., 2018), landslide susceptibility (Marjanović et al., 2011; Ballabio et al., 2012), and digital rocks (Ma et al., 2012).

Random Forests and other tree-based methods, including gradient boosting, have gained increased attention with the implementation into scikit-learn (Buitinck et al., 2013). Similar to NN applications, RFs are applied to Automatic Seismic Interpretation (Guillen et al., 2015) with limited success. Seismological applications including localization (Dodge et al., 2016), event classification in volcanic tremors (Maggi et al., 2017) and slow slip analysis (Hulbert et al., 2018). Further geomechanical applications include fracture modelling (Valera et al., 2017) and fault failure prediction (Rouet-Leduc et al., 2017; Rouet-Leduc et al., 2018). Gradint Boosted Trees were the most performant models in the 2016 SEG ML challenge (Hall et al., 2017) for well-log analysis, propelling a variety of publications in facies prediction (Bestagini et al., 2017; Blouin et al., 2017;

Caté et al., 2018; Saporetti et al., 2018). Moreover, random forests were applied to detect reservoir property changes from 4D seismic data (Cao et al., 2017).

Furthermore, various methods have been applied to various doomains. Hidden Markov Models were used on seismological event classification (Ohrnberger, 2001; Beyreuther et al., 2008; Bicego et al., 2013), well-log classification (Jeong et al., 2014; Wang et al., 2017a), and landslide detection from seismic monitoring (Dammeier et al., 2016). KNN has been used for well-log analysis (Caté et al., 2017; Saporetti et al., 2018), seismic well ties (Wang et al., 2017b) combined with DTW and fault extraction in seismic interpretation (Hale, 2013b). The unsupervised k-means equivalent has been applied to seismic interpretation (Di et al., 2017a), ground motion model validation (Khoshnevis et al., 2018), and seismic velocity picking (Wei et al., 2018). The biologically inspired ant-tracking algorithm is commonly used for seismic interpretation (Pedersen et al., 2002) and in conjunction with NNs (Zheng et al., 2014). Graph modelling has been applied to seismology in modelling the earthquake parameters (Kuehn et al., 2011), basin modelling (Martinelli et al., 2013), seismic interpretation (Ferreira et al., 2018) and flow modelling in Discrete Fracture Networks (DFNs) (Karra et al., 2018).

Machine Learning methods have been applied to various disciplines in geoscience, with the main objective increasing predictive capability or automating expensive and labour-intensive tasks. These approaches rely on diverse labelled data sets and are prone to common problems of ML in geoscience inherent to geoscientific data and the implication of cost of data acquisition.

2.3.2 Challenges of machine learning in geoscience

Statistical methods and machine learning are based on several assumptions and demand some pre-requisites that can cause problems in geoscience. These include the assumption that data is independent and identically distributed (iid) and the pre-requisite of a ground truth for supervised learning. In this section I discuss these challenges and present some approaches to solve these problems.

Geoscientific data is known to be very heterogeneous across vastly different scales (mm to km), which makes the system hard to model in general. Additionaly, a core assumption of statistics iid is usually in conflict with the geological processes. Regionality of depositional patterns violates the assumption data is identically distributed and time-dependent processes, such as systems tracts in sedimentology, violate the independence assumption of individual samples. This fact has to be taken into account, when choosing models and sampling methods. Expanding on the sedimentology example, the time-varying deposition can be modelled as markovian (Schwarzacher, 1972), instead of treating samples as strictly independent. Moreover, sampling of any data needs to honour the clustering in distribution of samples. Stratified sampling (Kish, 1965) can alleviate sampling bias. Additionally, stratified sampling can address the problem that geoscientific data often contains imbalanced data. Imbalanced data implies that the number of samples per class in the label data set is not uniformly distributed. These imbalances can stem from the fact that different depositional regimes cause different

thicknesses in the stratigraphic columns, for example commonly leaving thicker sand columns and fine shale layers. Alternatively, imbalances can stem from the data collection process itself, be it that seismic data does not adequately image variations below 10 m or the location where data is collected, considering that e.g. E&P companies do not choose the location for 3D seismic data acquisition randomly. This imbalance due to non-uniform sampling can not be solved by sampling itself, as the bias is implicit in the available data itself.

In the computer vision community hand-labelled data sets like ImageNet, CIFAR, and PASCAL-VOC are openly available, which catalyzed the developed new architectures and approaches in deep learning. Geoscientific data is often expensive to acquire and companies are reluctant to make data available, less even for processed or interpreted data. Early machine learning workshops often showed results on the open Dutch F3 dataset, however, national data repositories have started to change this approach to foster innovation. With data becoming more available, the next problem is the lack of ground truth. Obtaining accurate labels for seismic data is impossible, as any inversion process is non-unique and digging is not practical. In other imaging-based fields (e.g. radiology) that rely on interpretation of imaging results, studies investigate both inter-interpreter variations, by making several interpretations available and intra-interpreter variability by re-interpreting the dataset after a set time interval (McErlean et al., 2013; Alikhassi et al., 2018; Al-Khawari et al., 2010). Additionally, simulations provide a ground truth, but can implicitly include modelling assumptions in the data or commit the inverse crime (Wirgin, 2004). The inverse crime presents the problem of modelling and inverting data with the same theoretical ingredients.

In geophysics itself, seismic data presents a unique challenge to computer vision problems, in that the 3^{rd} percentile of amplitudes occupy large parts of the dynamic range (Forel et al., 2005). Displays of seismic data usually clip amplitudes to make most of the seismic amplitude content visible, this has also proven to be a viable preprocessing step before feeding seismic data to computer vision systems, such as convolutional neural networks. Machine learning systems have been known to be vulnerable to noise. This noise can be physical noise (i.e. low SNR) for simpler models or adversarial attacks that reverse engineer more complex models to fool said model. Adversarial attacks include a one-pixel attack on ImageNet classifiers (Su et al., 2019), humanly imperceptible noise (Goodfellow et al., 2014a), or physical stickers (Brown et al., 2017). In addition, geological data contains regions of geological interest and regions that are inconsequential, this has not been represented in metrics adequately (Purves et al., 2019).

Realistically, the sparsity of labelled ground truth data can be addressed in different ways. In the case when labels are available but not abundant, transfer learning of highly generalizable models like VGG-16 can be fine-tuned to seismic data. The VGG-16 architecture can also be included in U-Nets as a decoder to leverage the benefits of transfer learning in semantic segmentation tasks (Dramsch et al., 2018c). Moreover, weakly-supervised training can preform label propagation of labeled sections of the full data set to unlabeled sets. Unsupervised or self-supvervised training can be applicable, where no reliable ground truth is available, but a desired operation on the data is known or an internal structure of the data can be exploited (Dramsch et al., 2019b). Addition-

ally, multi-task learning has been shown to be able to stabilize network performance in Natural Language Processing (Liu et al., 2019) and Reinforcement Learning (Yu et al., 2019).

One caveat of increasingly performant but complex machine learning models is stakeholder buy-in or trust. These issues can be adressed, by benchmarking complex models against simpler models and physics-based solutions. Additionally, model explainability has become an important topic of research (Lundberg et al., 2017). Ribeiro et al. (2016) introduce the local interpretable model-agnostic explanation (LiME) method to gain insight into black-box models for individual samples. Shrikumar et al. (2017) propose a method to propagate activations in Deep Neural Networks. The Grad-CAM algorithm (Selvaraju et al., 2017) provides attention-like explanations for CNNs in computer vision tasks, to explain the main contributors to a classification output. Additionally, strict adherence to train-val-test splits and exploration of biases within the data can be essential. Considering these caveats and best practices in Machine Learning for geosciences, the following chapter introduces the main chapters of this thesis.

CHAPTER 3

Synopsis

The following chapters are comprised of four journal papers that are supplemented with two conference papers and four workshop papers, of which all are peer-reviewed or submitted to peer-reviewed journals. I combine several papers into topical chapters for conciseness. This thesis follows a data science workflow, starting with exploratory data analysis to gain insight to the specific geology and 4D seismic. I go on to present groundwork on machine learning and data processing on 4D seismic data of the Danish North Sea. Based on this groundwork, I developed a method for 4D seismic inversion and a novel unsupervised 3D time-shift extraction method for 4D seismic. This chapter summarizes these papers and places them in the appropriate context for the thesis.

3.1 Data Preparation

In Chapter 4 I include one published journal paper (Aabø et al., 2020), one published conference paper (Aabø et al., 2017), and one published workshop paper (Dramsch et al., 2018a). The published conference paper with the title "Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea" (Aabø et al., 2017) contains preliminary work contributing to and extended in the journal paper "An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field" (Aabø et al., 2020). Whereas, the workshop paper with the title "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk" (Dramsch et al., 2018a) is an independent study of backscatter scanning-electron microscopy data on chalk thin slices.

The data for this thesis was acquired in the Danish North Sea. The main hydrocarbon reservoirs in the Danish Central Graben area consist of chalk, a sedimentologically distinct feature in the seismic data. The chalk layer is high in porosity (20-35%), however, very low in permeability 3 mD to less than 1 mD. In Aabø et al. (2017) we presented an integrated fracture study of the Ekofisk chalk Kraka field in the South Central Graben. Within this preliminary study, we performed a localized fracture study along one wellbore to compare fracture measurements from core, well logs and seismic data. Initial analysis of the seismic data showed a maximum vertical resolution of ~40 m, which did not yield sufficient results for comparative study.

Figure 3.1 contains several post-stack seismic attributes to enhance lineaments within the seismic cubes. While the variance and structural cubes yielded some initial promise the following image processing workflow yielded the best results. These were geared to-

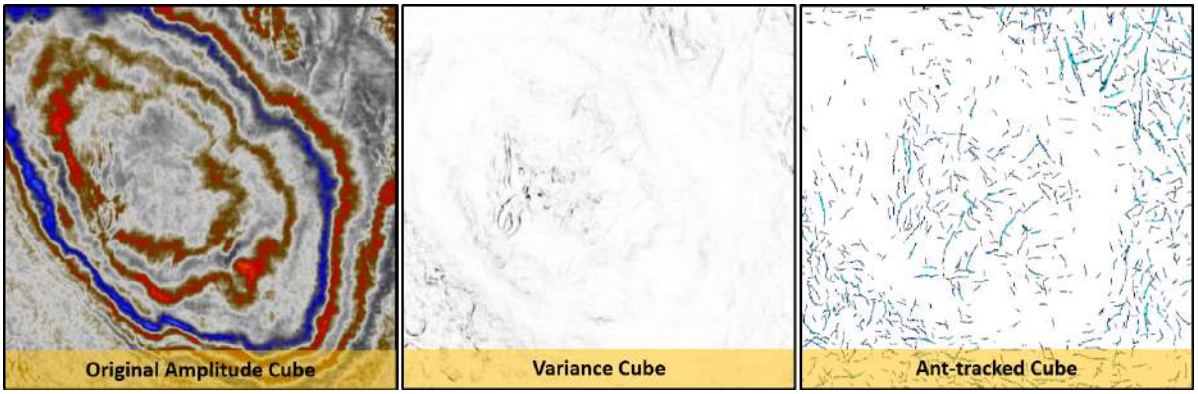


Figure 3.1: Comparison of seismic data, variance, and ant-track time slice to enhance fractures in images (modified from Aabø et al., 2020).

wards enhancing vertically coherent structures. This was achieved by a workflow including colorspace transformations and ant-tracking, which is a search algorithm leveraging biologically inspired software agents.

Normally, images are shown in Red-Green-Blue (RGB) colorspace, however, these can be transformed into other space, such as, Cyan-Magenta-Yellow-blacK (CMYK) and Hue-Saturation-Value (HSV). The HSV colorspace is commonly used in image analysis to detect edges on the gradient of the saturation values. Therefore, it serves as a good target colorspace for image processing. To achieve this, the bit-depth of post-stack seismic data (8 / 16 bit) has to be increased, considering that natural images displayed on modern monitors contain 32 million colors with a bit-depth of 24 bit for color representation. This is achieved by replicating the seismic cubes with a static timeshift to create an RGB representation (Laake, 2014). In our case a small shift below 3 ms to avoid loss of small-scale fractures and avoid smearing yielded the best results.

Consequently, after a colorspace transformation to HSV, the biologically inspired ant-track algorithm was applied to the saturation gradient volume. The ant-tracking algorithm implements unsupervised software-agents that search the vicinity in a 3D volume to find spatially coherent features (Dorigo, 1992). The software agents can be instructed to be more or less aggressive in their search, which provides a trade-off between better fault vertical enhancement or nuance of the smaller fractures. The workflow for fault extraction is shown in Figure 3.2.

Joint interpretation of the seismic and ant-track volumes yielded a focused seismic interpretation along the well-bore, where Borehole Imaging (BHI) data were available for comparison (after the interpretation to avoid bias). These match the independent interpretation of the well data closely in orientation and distribution of fractures. It is likely that these represent fracture corridors, small faults or damage zones in the chalk. This preliminary study was able to show that seismic provides a valuable method for mapping the size, orientation and connectivity of fracture zones away from the well.

Following this initial study, the seismic interpretation was extended for to regional fault systems and BHI to several wells for Aabø et al. (2020) presented in Section 4.2.

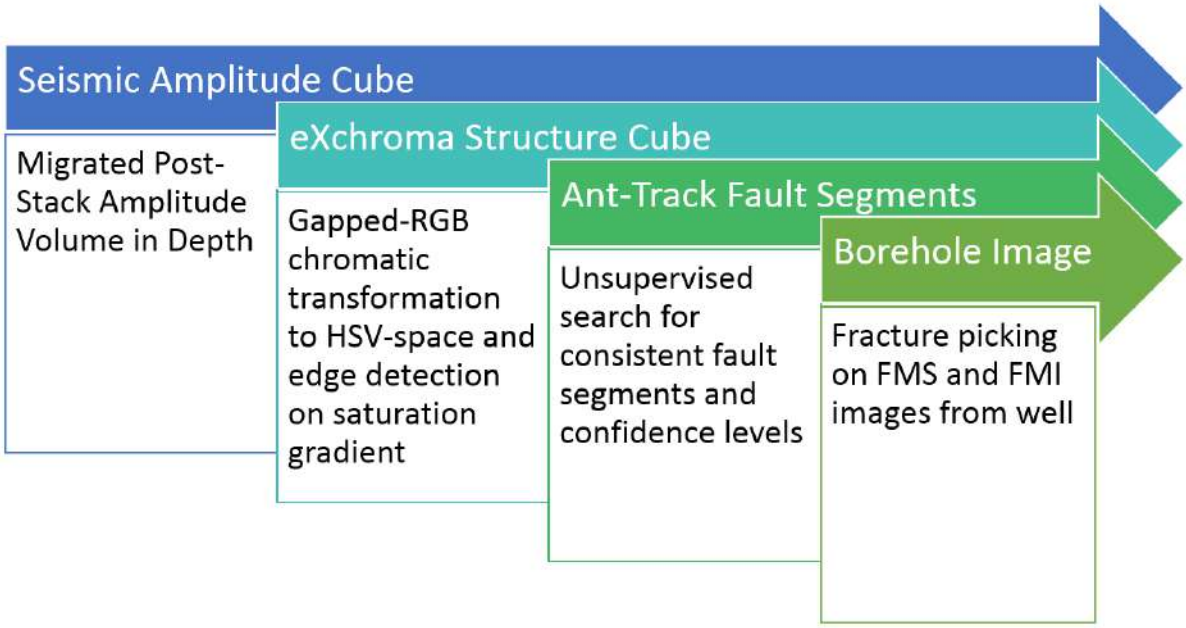


Figure 3.2: Workflow to identify fractures in post-stack seismic data to prepare for comparison to BHI analysis.

The analysis of ant-tracked attribute volumes, allowed us to relate structural trends below the resolution of amplitude seismic to features at different scales. This interpretation suggests that the fracture pattern is more complex than previously suggested. We propose that fracture generation and propagation in the field is in part controlled by the regional maximum horizontal stress from the seismic interpretation in addition to the halokinesis in the South Central Graben.

The seismic analysis was correlated with the fracture analysis from BHI data and core analysis in multiple wells. The fracture analysis was particularly dependent on the Terzaghi-correction (Terzaghi, 1965) to obtain the in-situ fracture orientation. This work identified two main fracture trends in the Danian Ekofisk reservoir. The main fracture set strikes sub-parallel to the regional NE/NNE maximum horizontal stress present on all horizontal/deviated wellbores and core. The vertical fracture distribution of the Kraka Field studied in a single well, due to availability. Within this single well the NE/NNE fracture distribution was continuous. This main NNE/NE fracture trend was traced from well scale to ant-tracked scale bridgeing the scale-gap. Regional large-scale faults interpreted on the raw amplitude seismic are present in the ant-tracked cube, trending NE, which indicates that the NNE trend is representative for smaller-scale lineations, with a Northern deviation on regional scales.

Further research into the porosity and sedimentology of the chalk reservoirs conducted on microscoping scales focused on identifying porosity using Backscatter Scanning-Electron Microscopy (BSEM) in Section 4.3, which comprises the third paper in this chapter (Dramsch et al., 2018a). Identifying the grain size and orientation of the oolites

is usually a manual work-intensive task, ideal for computer vision tasks, considering the good contrast of light-grey to white oolites and the black background. Unfortunately, training data was not available, so unsupervised clustering was appropriate to find the optimal boundary of the grains. Gaussian Mixture Models learnt a two-fold representation that separated the background well from the rock. Any single-valued decision boundary will be non-smooth, which can be alleviated by morphological filtering. Smooth boundaries are essential for chalk grains, as the perimeter of the oolites can be used to calculate the specific surface of chalk. The optimal boundary of chalk grains could then be used to generate training data for more sophisticated machine learning systems.

In the first study of this chapter (Aabø et al., 2017) we applied an image processing workflow to enhance the vertical resolution of seismic data. Consequently, The data was transformed to deploy a biologically inspired software algorithm to enhance and identify lineaments for better interpretation. This work was essential in enabling a localized pilot study to relate localized features from well-scale BHI to enhanced seismic-scale, verifying the seismic image analysis. This pilot study fed into a larger study (Aabø et al., 2020), where a fracture study from core and BHI was related to both the enhanced ant-tracked volume and a regional fault interpretation, updating the understanding of the fracture generation in the Salt Dome Process in the Danish South Central Graben area. The third study solved a manual process using an unsupervised image analysis tool paired with strong data science principles, providing a novel reliable tool to geologists in BSEM analysis.

3.2 Foundational Research

The foundational research in this thesis includes publications on Deep Learning and 4D seismic in Chapter 5. These publications apply a signal processing-approach to both 4D seismic and Machine Learning. I include a paper that takes a tutorial-view of dynamic time-warping a 4D seismic time shift analysis tool and introduces a novel constraint to improve performance of the algorithm. I then go on to present a possible source of misclassification in neural networks on non-stationary physical data such as seismics. I further investigate a possible solution to the aliasing problem of Convolutional Neural Networks for seismic, including complex-valued operations withing the network. I further investigate the assumption that massive interpreted datasets have to be available for successful training of Deep Neural Networks and present a working solution for smaller datasets.

Dramsch et al. (2019a) presents a tutorial of Dynamic Time Warping (DTW). DTW is a powerful signal processing tool introduced to 4D seismic analysis by (Hale, 2013a) on synthetic data. 4D seismic data relies on alignment of the seismic volumes. This enables interpreters to compare the amplitudes differences of the data. Due to the capability of DTW to match arbitrary time-series, it is applicable to 4D time shifts, seismic-well ties, well-to-well ties, and seismic pre- and post-stack migration (Luo et al., 2014). DTW

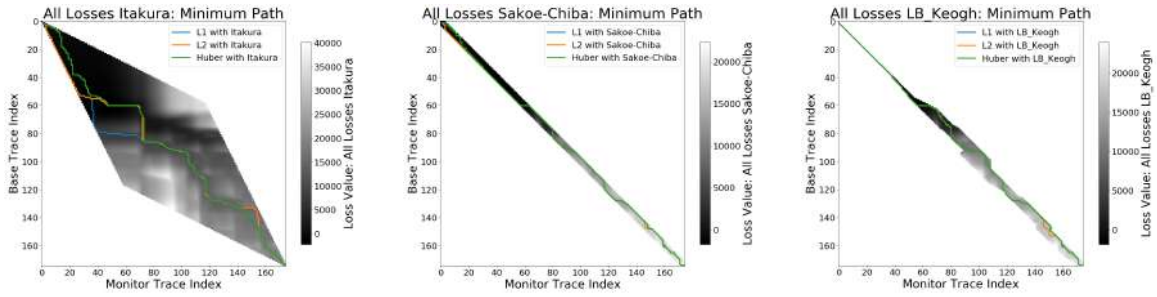
is known to be computationally slow and expensive, while extracting poor matches on seismic field data. This tutorial paper goes into detail of the DTW algorithm, exploring similarity measures, optimization, and constraints interactively through reproducible implementation in Python.

The DTW algorithm, represented in Algorithm 1, relies on calculating a distance matrix sample-wise between two traces. This is the first avenue of optimization we explore in this paper. The commonly used L_1 norm to calculate the distance norm is shown to perform worst out-of-the-box calculating $|b - a|$. Alternatively, the euclidean distance or L_2 norm can be used, which modifies the calculation to $(b-a)^2$. The difference between L_1 and L_2 is significant in the sense that the L_1 norm is not differentiable or convex, however it scales linearly for outliers. The L_2 norm converges fast close to zero, however the error "explodes" for outliers. We introduce a constraint used in convex optimization, which combines the advantages of the L_1 norm and L_2 norm, namely the Huber loss:

$$L_\delta(a, b) = \begin{cases} \frac{1}{2}(b-a)^2 & \text{for } |b-a| \leq \delta, \\ \delta(|b-a| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases} \tag{3.1}$$

which is convex for small values, scales linearly for outliers and is differentiable for all values of $\mathbb{R}$, with δ being a scaling factor.

Additionally, the search space on the cumulative distance matrix can be constrained to both increase performance and avoid non-optimal solutions. The different contraint strategies are presented in Figure 3.3. The Itakura parallelogram (Itakura, 1975) in Figure 3.3(a) describes a parallelogram that that has the largest width agress the diagonal of the matrix, providing the most flexibility for the DTW algorithm in the center parts of the seismic traces. The Sakoe-Chiba disc (Sakoe et al., 1978) follows a different strategy, which provides a constant maximum warp path. This strategy in Figure 3.3(b) introduces a global maximum time shift. Contrary to these two global constraints, we



(a) Itakura (1975) Parallelogram

(b) Sakoe et al. (1978) Disc

(c) LB_Envelope (Keogh et al., 2005)

Figure 3.3: Minimum path for constraint masks for cumulative cost in DTW. Images show the optimum path for different loss functions L_1, L_2, and Huber loss (from Dramsch et al., 2019a).

procedure DTW(a, b)
  Given: Trace a and Trace b of lengths n.
  function CALCULATE DISTANCE MATRIX $D(a, b)$
    $D \leftarrow dist(a, b)$
  end function
  function CALCULATE CUMULATIVE COST $C(D)$
    $C[0, 0] \leftarrow 0$
    for $i = 1$ to n **do** ▷ Populate Edge
      $C[0, i] \leftarrow D[0, i] + C[0, i - 1]$
      $C[i, 0] \leftarrow D[i, 0] + C[i - 1, 0]$
    end for
    for $i = 1$ to n **do** ▷ Fill Cumulative Cost Matrix
      for $j = 1$ to n **do**
        $C_{min} \leftarrow \mathbf{min}\{C[i, j - 1], C[i - 1, j - 1], C[i - 1, j]\}$
        $C[i, j] \leftarrow D[i, j] + C_{min}$
      end for
    end for
  end function
  function BACKTRACK MINIMUM COST PATH $P(C)$
    $P \leftarrow C[n, n]$
    while $i > 0 | j > 0$ **do**
      $i, j \leftarrow \mathbf{index}\{P[last]\}$
      $C_{min} \leftarrow \mathbf{min}\{C[i, j - 1], C[i - 1, j - 1], C[i - 1, j]\}$
      $P.\mathbf{append} \leftarrow \mathbf{index}\{C_{min}\}$
    end while
  end function
  return P
end procedure

Algorithm 1: Dynamic Time Warping algorithm consists of calculating the element-wise distance matrix, cumulative cost and then find the optimal path in the cumulative cost matrix.

introduce the LB_Keogh (Keogh et al., 2005) constraint in the paper. This lower bounding method provides a mathematical lower bound for the DTW algorithm. We use this lower bound to constrain the warp path, which provides larger variability to high amplitude areas, where cycle-skipping can occur, presented in Figure 3.3(c). The results of combining the Huber loss with the LB_Keogh constraint are presented in Figure 3.4.

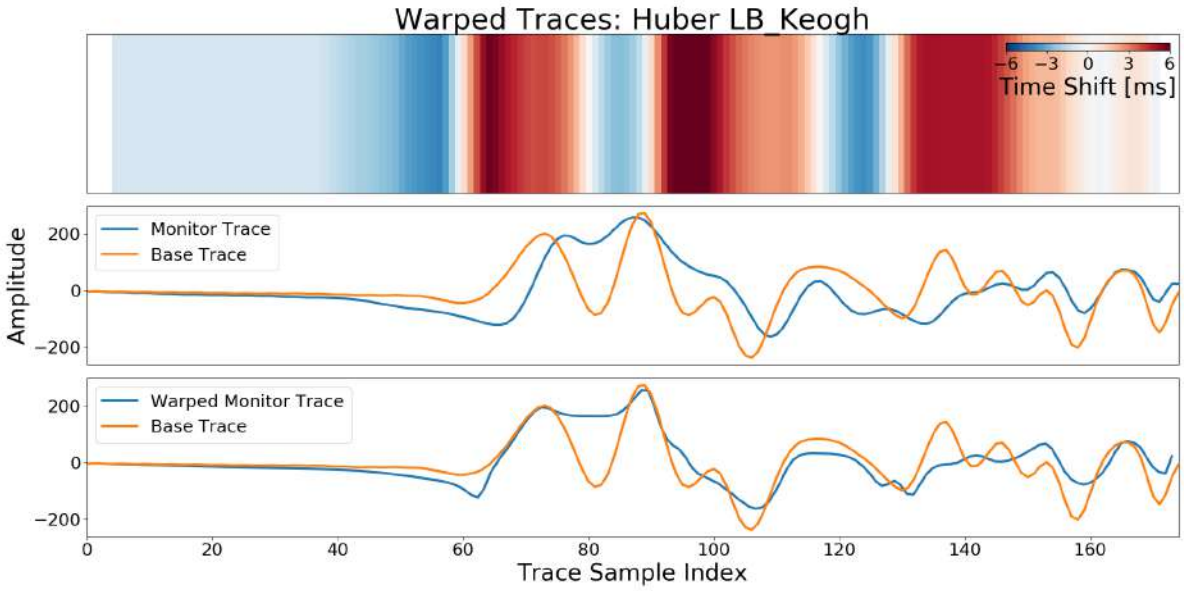


Figure 3.4: Time shifts and warped traces (from Dramsch et al., 2019a).

In the workshop paper "Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series" (Dramsch et al., 2018d) the insight from applying the LB_Keogh constraint was transferable to CNNs. CNNs apply a windowed convolution to the input data. Windowed areas of non-stationary physical data can be offset from the usually baselevel of an amplitude of zero. In the case of seismic data, traces tend to be zero-centered. In the case of a simple activation of a single neuron in a NN with $\sigma(w \cdot x + b)$ (cf. Section 2.2.2), this equates to a bias of $b = 0$. Seismic data contains sections that fall within the range of most patches, where the reflection response is entirely offset from zero, which equates to a mean-shift within the network. This paper served as a preliminary study for Dramsch et al. (2019g), which explores a solution for this property of patch-based training in CNNs that deteriorates the generalization.

Neural Networks apply real-valued transformations on the data, discarding phase information entirely. Figure 3.5 shows the spectra of the full trace in green as a background, with two selected cutouts of different sizes overlaid. It is clear that both windows show a sufficiently good reconstruction of the original amplitude spectrum, except for the offset at the low frequencies. The slope of the phase spectrum is reconstructed somewhat by the larger window, but non can reconstruct the notch. Many deterministic signals contain significant information in the phase of the signal. Discarding the phase

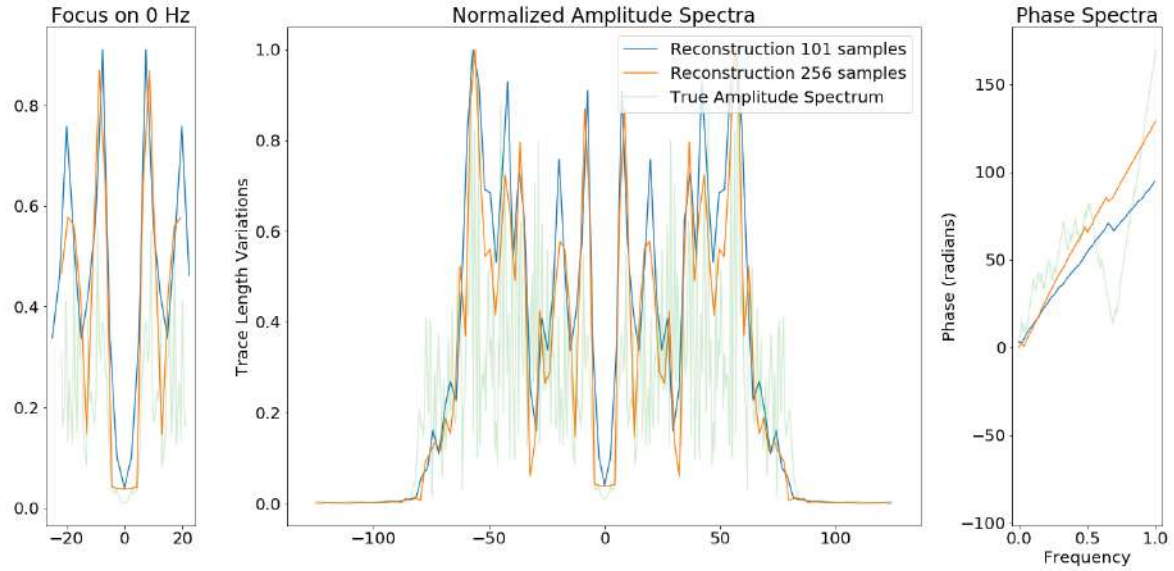


Figure 3.5: Normalized spectra of windows of trace with "offset" zero. Aliasing of the low frequencies is visible.Phase information not reconstructed from windowed data, slope depending on the window size. Data tapered before FFT (from Dramsch et al., 2018d).

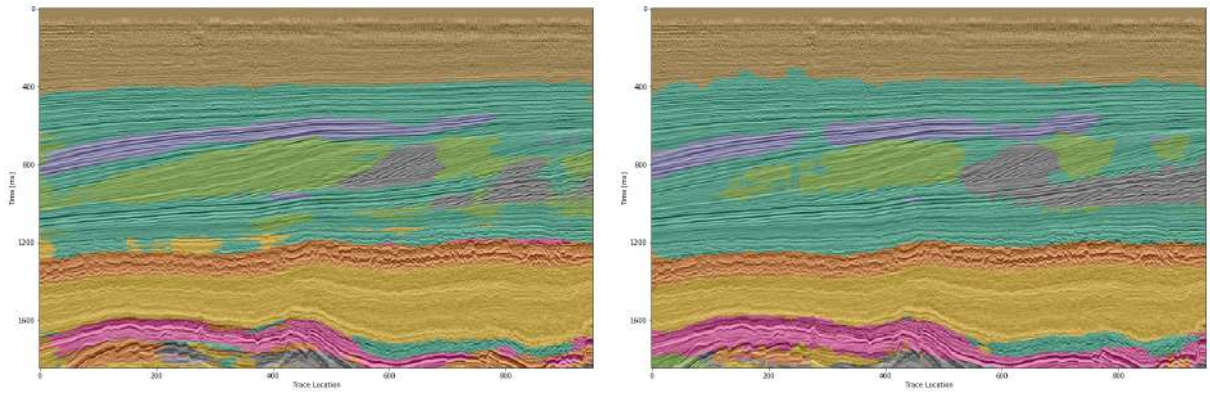
information leads to low-frequency aliasing analogous to the Nyquist-Shannon theorem for high frequencies.

In the paper "Complex-valued neural networks for machine learning on non-stationary physical data" (Dramsch et al., 2019g) I explore complex-valued deep convolutional networks to leverage non-linear feature maps and show that in non-stationary data, the phase content improves generalization of CNNs. Furthermore, complex-valued networks result in a smaller network with better performance compared to a larger real-valued network. In this study I implemented a deep convolutional AutoEncoder to compress 2D slices from a 3D seismic cube to evaluate the reconstruction error. There is a difference of network implementations, where complex-valued neurons are represented as two feature maps, one for the real component and complex component each. Therefore, matching the networks proved to be a complicated task, which led me to build four different architectures that get progressively bigger and compare the results.

The work in "Complex-valued neural networks for machine learning on non-stationary physical data" (Dramsch et al., 2019g) was in part based on reconstruction to test lossy compression and reconstruction of seismic data. Another reason to implement an unsupervised method was the limited availability of realiable interpretations of seismic data. Defining a decision boundary for seismic interpretation is only in the beginning stages of research, which leads us to the decision to inspect reconstructed seismic numerically as signal analysis is well-explored in seismic data processing. Therefore, analysing the result in the Frequency-Wavenumber (FK)-domain was possible and gave additional insight to

the denoising effect of the AE.

Nevertheless, some interpretations are available openly and companies often have a plethora of interpretations and re-interpretations of seismic data, making automatic seismic interpretation a topic of interest as evidenced by Table B.1. However, Deep Neural Networks are notorious for needing large numbers of diverse annotated samples. That is often prohibitive to geoscience applications of Machine Learning. In "Deep-learning seismic facies on state-of-the-art CNN architectures" (Dramsch et al., 2018c) we show that state-of-the-arts Convolutional Neural Networks pre-trained on ImageNet can be transferred to perform Automatic Seismic Interpretation. Figure 3.6 shows the results of a fully trained network compared to a pre-trained network. The pre-trained network decreases both training time and data requirements significantly, while not compromising accuracy. A pre-trained network with diverse generalizable learned filters seems to alleviate some limitations of smaller non-diverse data sets used in the fine-tuning process.



(a) Waldeland CNN trained from scratch **(b)** Pre-trained VGG-16

Figure 3.6: Automatic Seismic Interpretation on two networks, trained from scratch and fine-tuned on pre-trained VGG-16 architecture. The pre-trained network generating a more consistent seismic interpretation, however showing an overall deficiency in diverse training data (from Dramsch et al., 2018c).

This chapter summarizes the foundational work conducted to enable the developments of concrete applications of Deep Learning in geophysics. These foundations touch on signal processing fundamentals in 4D seismic exploring metrics and constraints, then introducing a new constraint for Dynamic Time Warping in 4D seismics. The work in Deep Neural Networks includes an investigation into aliasing of patch-based training of Convolutional Neural Networks and including phase information in complex-valued neural networks. Finally, leading to an exploration of transfer learning for efficient training of deep learning models in Automatic Seismic Interpretation.

3.3 Machine Learning in 4D Seismic Inversion

A primary application of Machine Learning is building regression models. These regressions are suited for application in physical inversion problems, considering the value of priors in non-unique solution spaces. This chapter consists of two workshop papers that illuminate a DL solution approach from a network architecture analysis in the paper titled "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion" (Dramsch et al., 2019e) and a data perspective in the paper titled "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data" (Dramsch et al., 2019d).

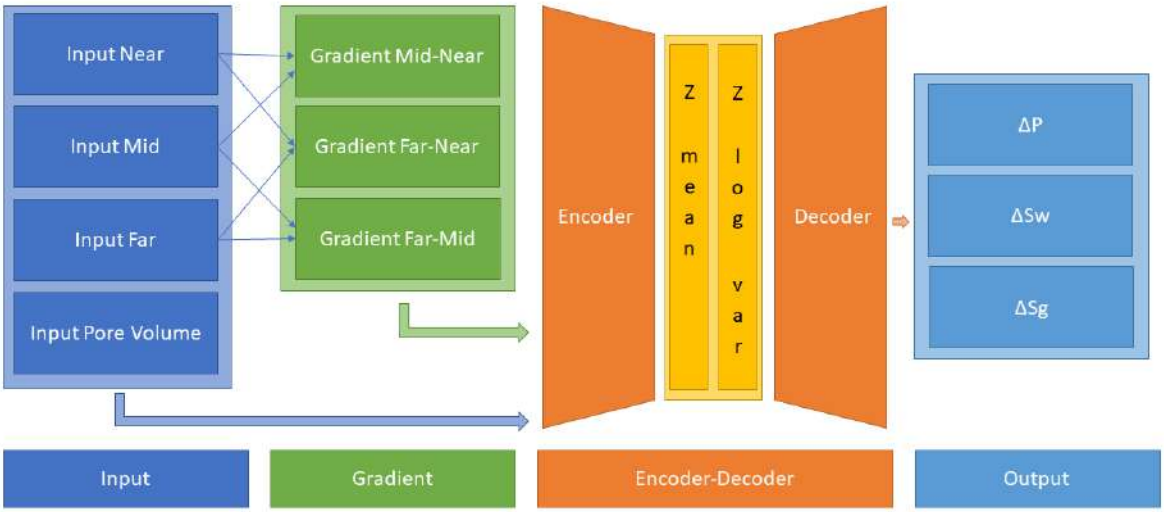


Figure 3.7: Architecture includes automatic physics-based gradient calculation of input seismic and an variational encoder-decoder architecture to invert seismic data for pressure and saturation changes (from Dramsch et al., 2019e).

Traditionally, 4D seismic Qantitative Interpretation (QI) often relies on priors to reduce variance in the face of uncertainty. The inversion problem in this chapter is a pressure-saturation inversion from seismic amplitude difference maps in the Schiehallion field. The Schiehallion field is a stacked turbidite reservoir in the UK North Sea, which makes it very heterogeneous and compartmentalized. The T31 sandstone reservoir has the most lateral extent with the thickness ranging from 5 m to 30 m. The small thickness of the reservoir layer results in the entire reservoir being contained in a single trough of a seismic wavelet, which leads us to treat the network as a 2D map instead of a 3D problem. The data available consists of simulation and field data with several timesteps of seismic data in near-, mid-, and far-angle stacks, and pore volumes, as well as, pressure changes and saturation changes for water and gas from simulation.

In Dramsch et al. (2019e) we present a novel network structure that explicitly includes AVO gradient calculation within the network as physical knowledge, shown in Figure 3.7. The network architecture was chosen to follow an encoder-decoder architecture as a forc-

ing function for information distillation. Additionally, the bottleneck layer implements a variational encoding layer to be less susceptible to noisy input. The network explicitly includes AVO gradient calculation in the network architecture, considering it is physical knowledge we know will stabilize pressure and saturation change separation. Including basic physics knowledge leads to the network learning residual information, essentially defining another forcing function for the networks learning process.

The initial phase was carried out on simulation data with a train test split, leaving a full 4D time step as validation set. Neural Architecture Search was applied to the network to determine depth and width of the architecture, using a Tree of Parzen Estimator (TPE) hyper-parameter search (Bergstra et al., 2015). Afterwards, to transfer the network to field data, the input of the network was combined with additive Gaussian noise (Bishop, 1995) to train the network for noisy field data input. This was a manual process of estimating good noise levels.

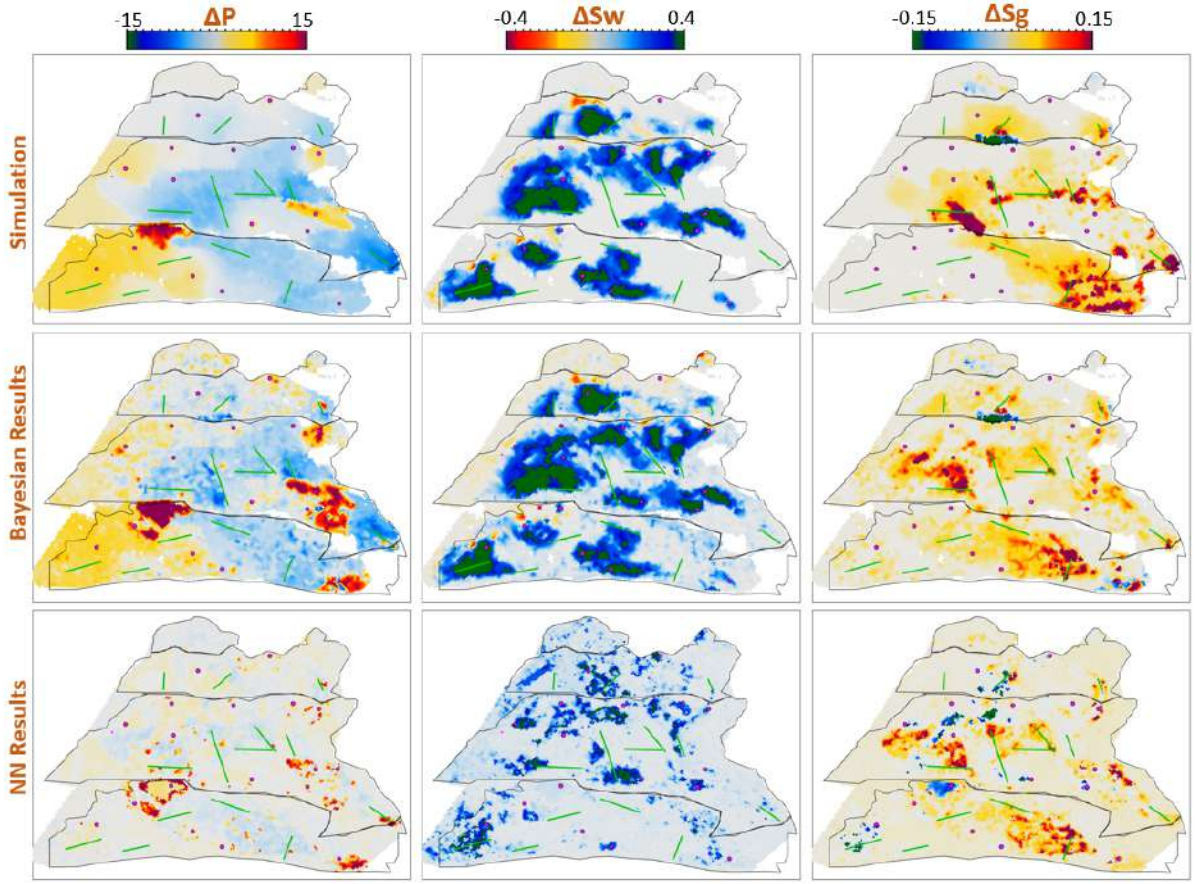


Figure 3.8: 4D QI inversion results from Bayesian inversion and Neural Network inversion. Bayesian inversion closely resembles simulation output. NN result showing good coherency, consistent amplitudes, but problems in strong changes of gas saturation (from Dramsch et al., 2019d).

The workshop paper Dramsch et al. (2019d) contains these results compared to the

simulation results and Bayesian inversion results, shown in Figure 3.8. These initial results on limited training data show that the stochastic process can extract pressure saturation information from field data, after training on simulation data. While the overall result is promising, regions of strong gas saturation changes present a problem. This could be contingent problems in the modelling as well as the fact, that they generate strong amplitude differences and are far in between, essentially behaving like outliers.

Learning meaningful information in deep neural networks is often contingent on interpreting the neural network. The results presented in Figure 3.8 contain three indicators that the network learned a meaningful inversion regression for the Schiehallion field. The network gets the overall trend in increase and decrease of pressure and saturation correct. Additionally, the range of output values for the network is unconstrained, but the network provides values in the ranges, that are expected from the simulation and Bayesian inversion results. However, and more interestingly, the networks does not contain spatial information, being a feed-worward DNN not a CNN, yet returns continuous albeit noisy outputs when assembled into maps.

This chapter comprised of two workshop papers, shows a working implementation of a machine learning system inverting pressure-saturation data from seismic. Moreover, an implementation of a network trained on simulation data that is transferred to field data by noise modelling is presented. Finally, we show that including basic physics in the network architecture stabilizes training, making the case for physics-based Machine Learning. Two journal papers are in preparation but not included in this thesis that analyze the network structure and the training data in detail (Corte et al., 2019; Dramsch et al., 2019f).

3.4 Machine Learning in 4D Seismic Time-Shift Extraction

This final chapter consists of the submitted journal paper “Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks” (Dramsch et al., 2019b). This paper presents a novel 3D warping technique for the estimation of 4D seismic time-shifts. The algorithm is unsupervised and provides 3D warp-fields with uncertainty measures, while avoiding many limiting assumptions.

4D seismic time shift extraction is often done in 1D, due to time constraints and often sub-par performance of 3D algorithms. This chapter explores and summarizes conventional 3D warping methods and machine learning approaches. Many of these algorithms rely on classical cross-correlational or optical flow approaches. Correlation-based algorithms can be susceptible to noise and inversion-based algorithms can take weeks to provide results and optical flow approaches suffer from the implicit assumption in standard implementations. These approaches suffer from the same limitations in Machine Learning systems just like conventional algorithms. In this chapter the medical Voxelmorph algorithm is adapted to match 4D seismic data volumes in 3D.

The Voxelmorph algorithm is based on the diffeomorphic assumption, which at its core describes the map of one data set to another data set, providing this map with particular properties. The main benefit of applying diffeomorphic mapping to geoscience data comes in the fact that all diffeomorphisms are homeomorphic. The homeomorphic assumption transfers well to the geological reality that the mathematical topology stays constant, resulting in reflectors neither crossing nor generating loops.

The algorithm is trained in an unsupervised, or rather self-supervised way to avoid the bias from time shifts that were extracted from any other method. Supervised training is discussed in the paper as implicitly introducing the assumption of the extraction method for the training data into the newly trained network.

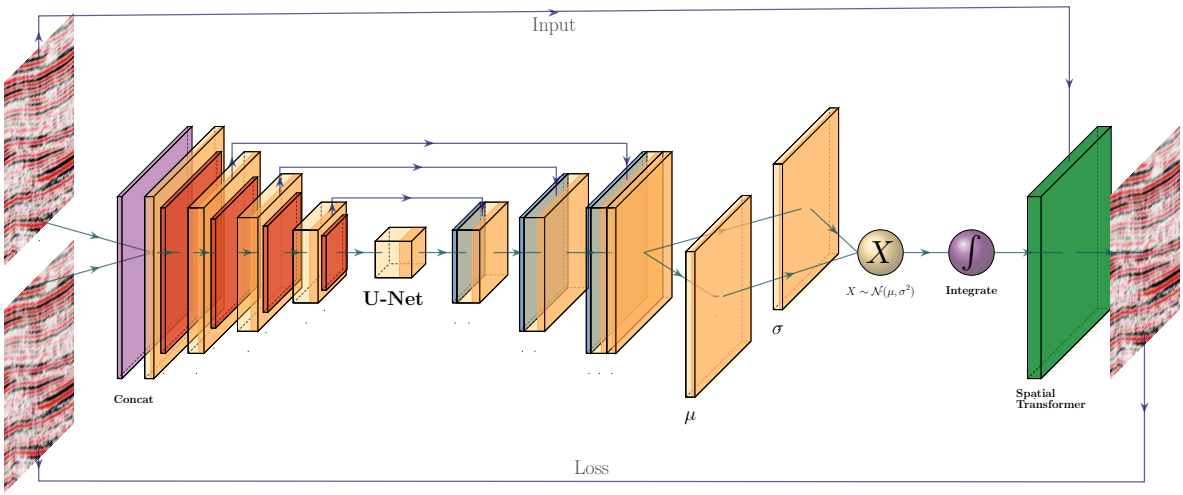


Figure 3.9: Voxelmorph Architecture 2D abstraction. Two 3D volumes are passed to the network, concatenated (purple) and passed to a U-Net architecture. The U-Net outputs two cubes that generate the mean static velocity and the standard deviation, which is sampled during training. The sampled value is integrated to obtain the diffeomorphic warp velocity used in the spatial transformer layer (green). The network evaluates losses on the KL-divergence at μ, σ and MSE between the warp result of the monitor volume and the warped base volume, enabling self-supervised training (from Dramsch et al., 2019b).

The architecture in the network uses the U-Net architecture to input two 3D seismic volumes and extract a static warp velocity field, shown in Figure 3.9. The static velocity field is extracted as a Gaussian distribution to measure the co-variance and provide uncertainty value of the three-dimensional warp field. The neural network itself does not warp the seismic data, to increase transparency of the process. The architecture following the U-net samples the extracted velocity distribution and integrates this value to obtain the diffeomorphic flow. These values are passed to a dense 3D warping mechanism to enable the unsupervised training. The losses involved are a Kullback-Leibler (KL)-

divergence on the stationary velocity field and MSE on the difference between that warped monitor volume and the base volume.

In the paper we present the modified self-supervised Neural Network system and test the results on the training data itself and two generalization test sets. The first test set is on the same field but recorded at different times to the training set, ensuring a similar underlying geology, whereas, the second test set is taken from an adjacent field, recorded at different times, testing the full transfer of the trained network. We go on to test the original Voxelmorph architecture, which uses upsampled velocity fields and evaluate the results against our modified architecture, which uses the full flow field. Overall, this technique introduces a generalizable Deep Learning approach to extract 3D time-shifts with uncertainty measures from raw stacked 4D seismic data.

3.5 Contributions of this Study

This thesis contributes tangible Machine Learning applications in geoscience that solve real-world problems as well as work that contributes to the fundamental understanding of signal processing and Neural Networks in non-stationary physics.

The explorational work of this thesis validates the impact of image processing for enhancing the resolution of seismic data and automatic fault extraction. This work investigated the scale-gap between local Borehole Imaging and regional seismic data (Aabø et al., 2017; Aabø et al., 2020). The exploration of Backscatter Scanning-Electron Microscopy data introduced a novel unsupervised method to extract chalk grain boundaries from image data and shows the improvement of subsequent morphological filtering (Dramsch et al., 2018a). These methods reduce labor-intensive manual tasks, introducing varying degrees of automation in geoscience workflows.

The foundational work investigates low-frequency aliasing in Convolutional Neural Networks (Dramsch et al., 2018d) and goes on to show that phase information in complex-valued neural network can stabilize the reconstruction of compressed seismic data. The smaller complex-valued network outperforms larger real-valued networks, however, a very large real-valued network can implicitly learn partial phase information (Dramsch et al., 2019g). The paper touches on deficits of current metrics applied to geoscience and exposes a periodic dimming effect of frequencies from neural networks that should be further investigated, particularly in the context of aliasing. This paper led to the creation of the open source software package `keras complex` to enable complex-valued deep learning in `Tensorflow` (Manual in C). The code was available in an older framework (Trabelsi et al., 2017), which I consolidated for this study. I went on to package the code, make it easily installable and generate online documentation to enable researchers to generate their own studies from this work (Dramsch et al., 2019c).

The research in Dramsch et al. (2018c) showed that transfer learning can alleviate the necessity for large amounts of labeled data, by re-using a neural network on natural images. This study showed the generalizable networks can be transferred to seismic data and outperform smaller networks trained from scratch, the smaller network size being

necessary to avoid overfitting. The source code for this research was made available and has been of use to multiple researchers (Dramsch, 2018). This has wide applications in industry and research settings, considering the limited availability of labeled data and wide availability of pre-trained network architectures. Moreover, this insight is applicable to pre-training geoscientific Neural Networks and fine-tuning these models to specific applications when needed.

The tutorial paper (Dramsch et al., 2019a) contributes insights into applications of Dynamic Time Warping in 4D seismic. This work explores the influence of varying metrics in geoscience, introducing the Huber loss as a possible loss function for geoscientific application. The study goes on to show that a novel application of the LB_Keogh lower bound for Dynamic Time Warping can significantly improve the accuracy for field data, introducing a constraint on the search space for Dynamic Time Warping. The code for this tutorial including further interactive explorations into constraints for Dynamic Time Warping will be made available after the double-blind peer review process concludes (Dramsch, 2020).

The first application of Machine Learning to 4D seismic data introduces a novel method to perform pressure-saturation inversion on amplitude difference maps (Dramsch et al., 2019d). This work introduces basic physics principles into the neural network architecture, which was shown to stabilize the training result. Moreover, this work shows the possibility of training Deep Neural Networks on simulation data and subsequently transferring the network to field data, by applying adequate noise injection. The Deep Neural Network results were successfully compared to results from the Bayesian inversion showing a promising application of Deep Neural Networks in 4D Qantitative Interpretation (Dramsch et al., 2019d). While this work has attracted interest in a sponsors meeting and the workshop presentations (Dramsch et al., 2019d; Dramsch et al., 2019e), further investigation into model explainability and lower complexity baseline models is necessary (in preparation Corte et al., 2019; Dramsch et al., 2019f).

The second application of Machine Learning to 4D seismic data presents a completely novel method for time-shift extraction (Dramsch et al., 2019b). This method combines recent advancements in diffeomorphic mapping, Deep Learning and unsupervised learning to introduce a 3D time shift extraction method including uncertainty values, where 1D extraction is the standard. The method is shown to work on 3D seismic post-stack data with strongly differing acquisition parameters, without supplying any time shift information. After applying the method, the 3D seismic volumes are aligned well, with the diffeomorphic constraint performing well on seismic data. This work tests the trained network on two other 3D seismic volume pairs to test the generalization of the Convolutional Neural Network after training. The two test sets show that the trained model on a single 3D seismic volume pair transfers well to the same field with different acquisition parameters and even a different field with vastly different geological setting. The code is openly available to foster further development and investigation of the method (Dramsch, 2019).

Overall, this thesis shows the impact Deep Learning can have in geoscience with two novel methods for 4D seismic analysis. It shows the impact of fundamental research of signal processing and information theory in Deep Neural Networks and Convolutional

Neural Networks. Moreover, it shows applications of building systems in label sparse environments to leverage technological advancements and the value of including prior physical insights into Machine Learning workflows. This thesis builds heavily on open source software and aims to return some of the effort.

CHAPTER 4

Data Preparation and Analysis

This chapter is comprised of three accepted papers that make up the exploratory data analysis and introduction to image processing in seismic and unsupervised machine learning.

Papers:

T. M. Aabø, J. S. Dramsch, M. Welch, and M. Lüthje (2017). "Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea". In: *79th EAGE Conference and Exhibition 2017*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201701283. URL: https://doi.org/10.3997/2214-4609.201701283

T. M. Aabø, J. S. Dramsch, C. L. Würtzen, S. Seyum, F. Amour, M. Welch, and M. Lüthje (2020). "An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field". In: *Marine and Petroleum Geology* 112. Published, Chapter 4. ISSN: 0264-8172. DOI: https://doi.org/10.1016/j.marpetgeo.2019.104065. URL: http://www.sciencedirect.com/science/article/pii/S026481721930501X

J. S. Dramsch, F. Amour, and M. Lüthje (2018a). "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201803014. URL: https://doi.org/10.3997/2214-4609.201803014

4.1 Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea

Abstract: We present an integrated fracture study in the Ekofisk chalk reservoir of the Kraka Field, offshore Denmark, based on core, borehole images and seismic data. The core contains numerous fractures ranging from short (cm-scale) fractures, mostly associated with chert or stylolites, to large (m-scale) open, slickensided fractures likely related to halokinesis. On borehole images, especially larger fractures are identified, coinciding in dip and dip-azimuth. Seismic data at an approximate resolution of 40m would not resolve these local features around the well-bore. We show that chromatic analysis combined with an ant-tracking algorithm extracts several lineaments (> m-scale) from the seismic data. These correlate closely in orientation and distribution with the fractures logged in the well data. It is likely that these represent fracture corridors, small faults or damage zones in the chalk. The seismic data therefore provides a valuable method for mapping the size, orientation and connectivity of fracture zones away from the well. This gives insights into the scalability of local stress fields, and fracture distributions.

Key points:

- Borehole Imaging (BHI) essential to determine open vs cemented fractures
- Red-Green-Blue (RGB) processing enhances fracture resolution in seismic
- Ant-track algorithm automates fracture interpretation
- Fractures from enhanced seismic relateable to BHI and core

T. M. Aabø, J. S. Dramsch, M. Welch, and M. Lüthje (2017). "Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea". In: *79th EAGE Conference and Exhibition 2017*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201701283. URL: https://doi.org/10.3997/2214-4609.201701283

Introduction

The Kraka oil Field is a salt-induced anticlinal structure located in the southernmost tip of the Danish Central Graben (Danish North Sea). It is produced through natural depletion of the Danian Ekofisk Fm, an overpressured, naturally fractured chalk reservoir. Ekofisk chalk is a mono-mineralic carbonate rock that consists of 96 - 99% calcite ($CaCO_3$), non-carbonate biogenic particles and small amounts of clay particles (Abramovitz et al., 2010). The Danian of the Kraka Field is divided into an upper porous zone (units D1 – D3) and a lower tight zone (D4 – D5). Porosities in the porous zone are in the range of 25 - 35%, and vary only slightly across the field (Klinkby et al., 2005). In the reservoir, silica occurs as continuous chert bands and isolated chert nodules. The matrix permeability is less than 1mD, however, the effective permeability is approximately 20 times that due to the presence of fractures.

Tectonic fractures related to halokinesis are the main permeability enhancers. Smaller fractures associated with cherts and stylolites may however be important for local permeability enhancement. Fractures in Kraka occur in swarms (Jorgensen et al., 1991). Fracture spacing, orientations and connectivity in the field are currently not well constrained.

In this extended abstract we compare fractures and fracture zones observed at different scales on core, borehole images (FMS and FMI) and ant-tracked seismic volumes, and show that we can correlate between them. BHI and core data are highly complementary. Borehole images are cheaper, provide true orientations and survey the reservoir in-situ, so we can differentiate open and closed fractures under reservoir conditions. Cores allow for direct observations, analyses at nano and micro scale and laboratory experiments. Seismic is used to map faults and fracture zones away from the borehole and to identify regional structural trends. Combining the three data types allows us to extrapolate fractures away from the borehole, and will serve as inputs into a mechanically based discrete fracture model (DFN), which will improve future well planning and EOR activities.

Method and Theory

Micro-resistivity images from FMS and FMI tools are available in one vertical well and seven horizontal or deviated wells. The surveyed wells were drilled in the time period between 1989 and 1997 and raw data has been reprocessed for this study. Due to the high resistivity of chalk, caving and tool sticking, image quality is poor in many places. Chalk sections directly below chert bands have been particularly hard to resolve, as the chert bands do not have planar surfaces and so cause errors in the pad alignment stage of processing. The cherts, being highly resistive compared to the chalk, are in turn well resolved. A large portion of the fracture swarms in Kraka are chert associated. Additionally, cherts enable depth matching between borehole images and cores, as depth shifts along wells vary by up to 8 ft

Cores are available in three of the wells logged by BHI tools: Well 1 (deviated), Well 2 (horizontal), and Well 3 (vertical). This abstract focuses on analyses from Well 1, where the core-recovery percentage is highest. Relative fracture orientations measured in core have been reoriented, depth shifted and are plotted alongside image picks.

Fracture picks from BHIs and cores are subsequently compared to a structural framework derived from seismic images. Seismic amplitude cubes acquired in 2012 have a vertical resolution in the order of 40 m. Therefore, high-fidelity information from the Kraka amplitude cube has been extracted using a gapped chromatic method that is based on structurally sharpened satellite RGB/HSV processing (Laake, 2015). The resulting structural cube serves as an input for an algorithm that systematically analyze the data, mimicking the "swarm intelligence" of ants (Pedersen et al., 2005). The algorithm extracts structural lineaments and assigns confidence levels depending on the length and width of the path of segments, to enhance subtle compaction features and small-scale faulting that are essential to the interpretation of the Kraka chalk field. The additional structural information is used as an opaque overlay on the conventional amplitude cube to guide the seismic interpretation and to avoid misclassification of noise or acquisition artifacts. Centimeter- to meter-scale fractures identified in borehole images and cores are compared to these larger structural lineations in 3D.

Correlation of Core and Borehole Images

In the BHI data from the Kraka Field, sinusoids representing bedding (chalk and marl) are continuous across borehole images. Most fractures are however only represented by partial sinusoids, either because they are short or because they are only partially open or cemented. Comparison with core data, when available, is imperative in determining which partial signals should be picked. Lessons learned from cored wells are transferrable to BHI-surveyed wells without core.

The advantage of core is that we can identify smaller-scale stylolite associated fractures that are not detectable on images because the image resolution is about 1-2mm. Most open chert associated fractures are however visible, due to the large resistivity contrast between cherts and water-based drilling mud. The length of the chert-associated fractures depend on the size of the chert band or nodule, and varies between 10 and 50 cm in Well 1.

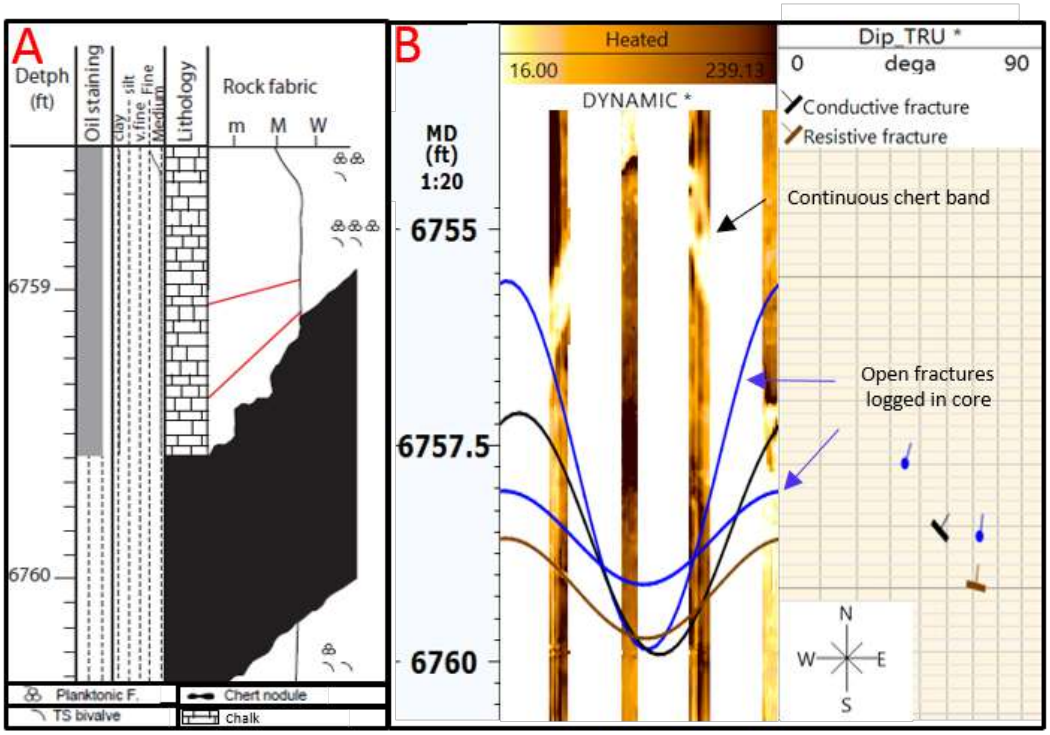


Figure 1 *Core log (a) with faults in red and Core-to-BHI correlation (b) of fractured interval in Well 1*

Figure 1 shows a logged core interval (a) with corresponding BHI section (b) from Well 1. The core interval contains two natural fractures, represented by blue tadpoles in the BHI. Both fractures are open (non-cemented) in the core, but are recognized as natural fractures by the presence of slickensides. The logged fractures coincide with one open (conductive) and one closed (resistive) fracture picked on the borehole image. The conductive fracture is associated with the continuous chert band, while the resistive feature is believed to represent a tectonic fracture. The relative timing of silica formation and saltdoming in Kraka is yet to be determined. However, as one of the closely-spaced fractures is cemented, while the other is not, it is reasonable to assume that they represent different fracture-generations.

The dip and azimuth of both fractures identified in BHI match the orientation of the fractures logged in core within 12^o. Small discrepancies are to be expected, as core must be reoriented manually to calculate true orientations, so fracture orientations from BHIs are commonly considered the most reliable, while the presence and type of fractures can be identified in the core.

Correlation of Seismic and Well Data

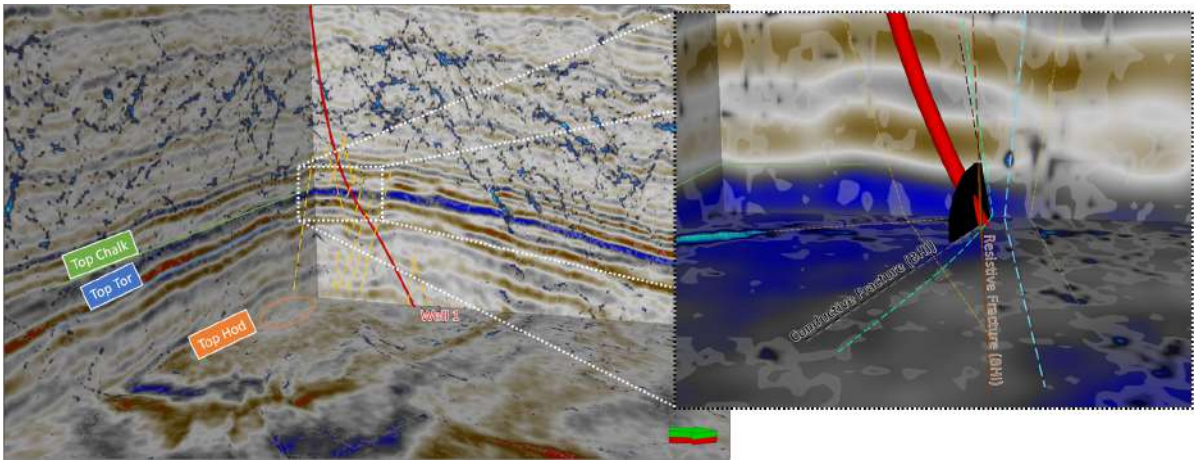


Figure 2 *Seismic overlaid with ant-tracked chromatic structural cube and close-up of interpretation compared to BHI fracture discs.*

In figure 2 we show the ant-track as transparent overlay over the seismic amplitude. The Top Hod to Top Chalk interval and the highly fractured overburden in the Kraka field are imaged. The z-plane in figure 2 cuts the inner chalk reservoir on Top Hod level. On this plane several lineations can be identified that strike radially relative to the anticlinal reservoir structure. The primary reservoir between Top Tor and Top Chalk shows several discontinuities and amplitude variations. On the cross-section we see conjugate inclined ($< 45^o$) faults cut through the reservoir and extend into the overburden and the underlying chalk package. Amplitude variations within the reservoir reflect compaction effects and possibly the influence of fluids saturation. The overburden is heavily fractured with low-throw (15m) conjugate faults that are highlighted with high confidence levels (blue) by the ant-tracking algorithm. The faults are parallel with a spacing between 50 m and 100 m and may be reactivated during depletion. This dynamic overburden must be corrected for in 4D seismic analysis.

The close-up in figure two highlights the faulting at reservoir depth. The z-plane was adjusted to reflect the middle Danian data. Cross-cutting faults along the wells are easily identified. High-confidence features along the well may be production related. The close-up also increases the visibility of lower confidence features from the ant-track algorithm. These also reflect low-throw conjugate faulting cutting the reservoir and compaction related features are highly visible in this display.

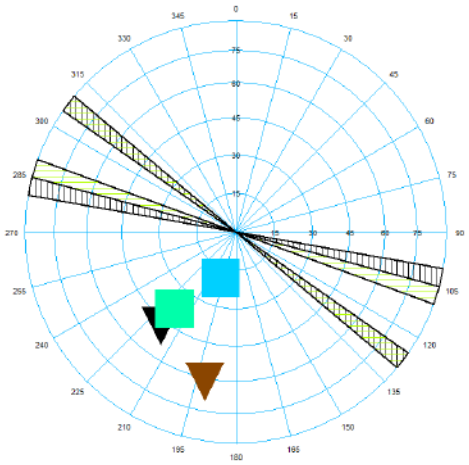


Figure 3 *Rose diagram with stereonet overlay, representing BHI picks as triangles and seismic fault as squares*

Along Well 1 cross-cutting faults are clearly visible. In the close-up, the two fracture picks from the BHI (Fig. 1) are imaged as discs corresponding to dip angle and azimuth. The Kraka reservoir is highly fractured and heterogeneities are clearly visible as amplitude variations on a seismic scale.

The dip and the dip-azimuth of the conductive fracture (black) corresponds well to the fault (green) from the seismic interpretation. The dip-azimuth of the resistive feature (brown) aligns with the seismic feature (blue), the dip angle deviates, however. The rose diagram in figure 3 shows the azimuth at a 5^o interval coinciding for both fractures. The stereonet overlay also confirms

the qualitative assessment that the dip angle of the resistive fracture is steeper than that of the turquoise fault plane and a good match of the conductive fracture with the blue fault plane.

These results show that seismic data and careful post-processing shows stress trends that are reflected at the subseismic scale by BHI and core data. Seismic data is not capable of distinguishing open and closed fractures. Parallel and conjugate faults (yellow) in figure 2 strengthen the case of a consistent regional stress field that scales down to local stresses observed at the BHI and core scale. These can serve as input to building a field scale DFN.

Conclusions

Integrated comparisons of core, borehole image and seismic structural data in the Kraka Field indicate that:

- Many of the fractures seen on core can also be identified on borehole images, especially chert associated fractures. However stylolite associated fractures identified in core are not visible on borehole images. Chert associated fractures, extensional fractures and faults are commonly represented by partial sinusoids in BHIs, suggesting they are either short or only partially open or cemented.
- Borehole images are imperative in distinguishing cemented and open fractures, and thus better constrain fluid flow along the fracture network.
- Chromatic method and ant-track algorithm allows us to image subtle faults, fracture zones and compaction features not obvious on amplitude cubes.
- Structural features picked on BHIs correlate to large-scale regional trends and to features picked on ant-tracked seismic data. This allows us to extrapolate them away from the wellbores and calibrate 3D models (e.g. discrete fracture network models).

This integrated study proves invaluable in testing assumptions in building fracture models and the subsequent upscaling process.

Acknowledgements

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding programme. We thank DUC for providing access to the data. We thank Frédéric Amour and Solomon Seyum for valuable discussion. We would also like to extend our thanks Schlumberger for providing licenses for Petrel, eXchroma and Techlog and ffA for the GeoTeric license, which were used in to interpret this data.

References

Abramovitz, T., Andersen, C., Jakobsen, F., Kristensen, L. and Sheldon, E. [2010] 3D seismic mapping and porosity variation of intra-chalk units in the southern Danish North Sea. In: *Geological Society, London, Petroleum Geology Conference series*, 7. Geological Society of London, 537–548.

Jorgensen, L., Andersen, P. et al. [1991] Integrated study of the Kraka Field. In: *Offshore Europe*. Society of Petroleum Engineers.

Klinkby, L., Kristensen, L., Nielsen, E.B., Zinck-Jørgensen, K. and Stemmerik, L. [2005] Mapping and characterization of thin chalk reservoirs using data integration: the Kraka Field, Danish North Sea. *Petroleum Geoscience*, **11**(2), 113–124.

Laake, A. [2015] Structural interpretation in color - A new RGB processing application for seismic data. *Interpretation*, **3**(1), SC1–SC8.

Pedersen, S.I., Randen, T., Sonneland, L. and Steen, O. [2005] *Automatic fault extraction using artificial ants*. 512–515.

4.2 An Integrated Approach to Fracture Characterization of the Kraka Field

Abstract: Oil and gas production of tight chalk reservoirs frequently rely on the presence of natural fractures, which increases the effective permeability of the reservoirs. Knowledge of these fracture systems can therefore be used strategically in well planning as well as in IOR and EOR efforts. Here we present an integrated workflow for fracture characterization in chalk, developed in the Kraka Field, located in the Danish sector of the North Sea. The workflow is based on data from borehole images, cores and seismic. By introducing two ant-tracked attribute volumes, which display structural trends below the resolution of amplitude seismic, we are able to correlate features at different scales. In Kraka, this approach has revealed that the fracture pattern is more complex than previously suggested. We propose that fracture generation and propagation in the field is in part controlled by the regional maximum horizontal stress and in part formed in response to salt movements.

Key points:

- Red-Green-Blue (RGB) processing bridges scale-gap between Borehole Imaging (BHI) and seismic
- Ant-track algorithm automates fracture interpretation
- Updated understanding of stress regime
- BHI fracture distribution can be extrapolated from well

T. M. Aabø, J. S. Dramsch, C. L. Würtzen, S. Seyum, F. Amour, M. Welch, and M. Lüthje (2020). "An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field". In: *Marine and Petroleum Geology* 112. Published, Chapter 4. ISSN: 0264-8172. DOI: https://doi.org/10.1016/j.marpetgeo.2019.104065. URL: http://www.sciencedirect.com/science/article/pii/S026481721930501X

Marine and Petroleum Geology 112 (2020) 104065

Contents lists available at ScienceDirect

Marine and Petroleum Geology

journal homepage: www.elsevier.com/locate/marpetgeo

Research paper

An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field

Tala Maria Aabø*, Jesper Sören Dramsch, Camilla Louise Würtzen, Solomon Seyum, Michael Welch

The Danish Hydrocarbon Research and Technology Centre (DHRTC), Technical University of Denmark, Elektrovej 375, 2800, Kgs. Lyngby, Denmark

ARTICLE INFO

Keywords:
Reservoir characterization
Fractures
Structural correlation
Borehole images
Core analysis
Seismic attribute volumes
Ant-tracking
Chalk

ABSTRACT

Oil and gas production of tight chalk reservoirs frequently rely on the presence of natural fractures, which increases the effective permeability of the reservoirs. Fracture characterization is therefore imperative in optimizing production schemes and obtaining economically viable recovery factors. Subsurface fracture characterization is often deemed challenging as the available data is typically of varying age and quality, and represents different scales. We have developed an integrated workflow for fracture characterization in chalk to address these challenges. The workflow is based on data from borehole images, cores and seismic. These data are typically available for most chalk (and hydrocarbon) fields. The interpreted borehole image dataset contains over 17 000 manual dip picks, ensuring a statistically viable base. A total of 150 m of core is available from 3 wells. The applied 3D seismic cube covers an 8 × 5 km hydrocarbon chalk field in the Danish North Sea.

In this workflow, the scale-gap between the data sets is bridged by the introduction of two ant-tracked attribute volumes, which display structural trends below the resolution of amplitude seismic. Further insight into the intricacy of subsurface fracture systems is obtained from fracture density logs, which provide an opportunity to study spatial distribution of fractures as well as a qualitative measure of fracture clustering. Cumulative density distribution plots and calculation of the variation coefficient of fracture spacing provide a more quantitative analysis of the fracture distribution.

The workflow, presented here in a step-by-step manner, is a general approach applied to data from the Kraka Field of the Danish North Sea. In the Kraka Field, the usage of this integrated approach shows that the fracture pattern in this region is more complex than previously suggested; probably controlled by the regional maximum horizontal stress and salt movements.

1. Introduction

Chalks typically represent high porosity - low permeability reservoirs, in which natural fractures are essential for hydrocarbon production (Koestler and Reksten, 1992). Knowledge of these fracture systems is often used strategically in well planning and in IOR and EOR efforts. Descriptions and models of natural fracture systems allow for simulation of flow and flow patterns in reservoirs, which in turn helps in understanding the quality and amount of hydrocarbon reserves. Natural fractures define the communication in reservoirs, which is the determining factor for well-placing decisions and setup of production schemes for IOR technologies (e.g. water flooding) and EOR technologies (e.g. smart water/chemical flooding).

Accurate predictions of natural fracture systems require an understanding of the controls on fracture orientations and distributions in an area (Fernø, 2012). We have developed a workflow to correlate structural features at different scales, based on borehole image-, core- and seismic data. The applied seismic data includes an amplitude volume and two ant-tracked volumes. High resolution lineations mapped on the two ant-tracked cubes (generated through a variance cube and through RGB-image processing of the 3D seismic volume, respectively) enables detection of smaller-scale lineations below the resolution of conventional seismic, thus bridging the scaling-gap between well and seismic data. Spatial distributions and fracture clustering is considered using fracture density data.

The suggested workflow has been applied to the Kraka Field. The Kraka Field, an asymmetric anticlinal structure located in the Danish Central Graben (Fig. 1), was chosen as a test-case as it is a relatively simple structure with a manageable amount of data. The applied data was provided by Maersk for the purposes of this research project. Kraka

* Corresponding author.
E-mail address: talama@dtu.dk (T.M. Aabø).

https://doi.org/10.1016/j.marpetgeo.2019.104065
Received 9 November 2018; Received in revised form 23 September 2019; Accepted 27 September 2019
Available online 04 October 2019
0264-8172/

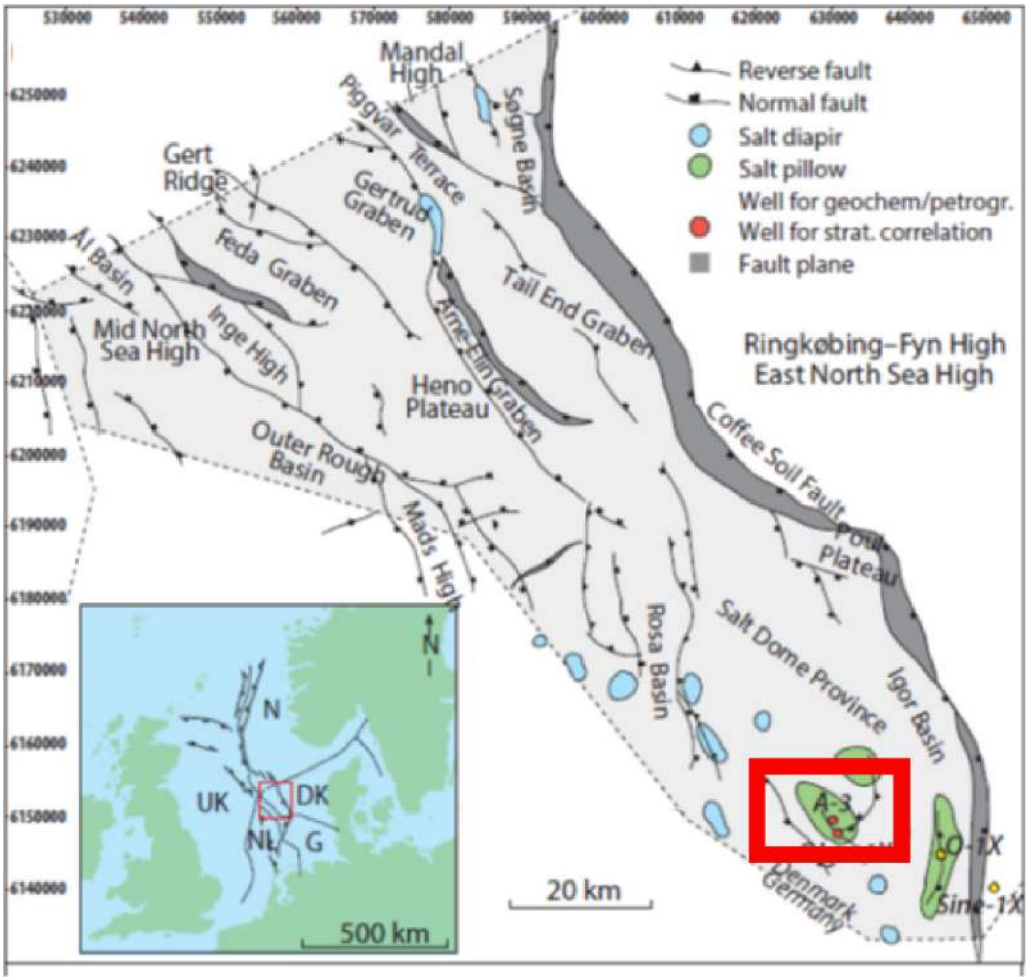


Fig. 1. Location of the Kraka Field, indicated by the red square, on a structural elements map of the Danish Central Graben (modified and edited from Møller and Rasmussen (2003)). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

is produced mainly from the Danian Ekofisk Formation but also from the Maastrichtian Tor Formation, both of which are naturally fractured chalk reservoirs (Jorgensen and Andersen, 1991).

Combining the three aforementioned data types in an integrated workflow allows for the extrapolation of fractures away from the borehole, increasing our holistic understanding of the natural fracture distributions, in this case of the Kraka Field. Using this integrated approach we are able to evaluate orientations of lineations from well-to seismic scale, allowing for structural modelling based on a geological conceptual model.

Our results will serve as inputs into a discrete fracture network model (DFN) founded on geomechanical principles for fracture propagation, which will improve future well planning and EOR activities in the area.

2. Geological setting

Prior to applying the workflow, the geological setting of the Kraka Field was considered.

The Kraka anticline was induced through halokinesis. Initiation started during the Triassic and continued to move during the remaining Mesozoic (Rank-Friend and Elders, 2004). The structure stretches more than 8 km along its long axis and approximately 5 km along its short axis (Rasmussen et al., 2005). The lithology in Kraka varies between pure chalk and marly chalk, with varying amounts of chert layers and nodules. Cores from the field show localized staining which is cyclic and often associated with the chert. On well-scale, three main structural features have been identified in Kraka. Large, open fractures with slickensides are abundantly observed throughout the core data. These fractures commonly terminate in clay rich layers. Smaller, chert-associated fractures occur frequently within the Ekofisk Formation and on occasion within the Tor Formation. Stylolite-associated fractures, which is predominantly observed in the Tor formation, are perpendicular to the pressure solution seems and are typically < 25 mm high.

Both reservoir units are characterized as tight. Porosities are in the range of 20–35% and permeabilities range from 3mD to < 1 mD (Klinkby et al., 2005). Effective matrix permeabilities in the field are significantly enhanced due to the presence of these natural fractures. The tectonic fractures (shear and extensional) are the main permeability enhancers. Smaller fractures associated with cherts and stylolites may however be important for local permeability enhancement (Jorgensen and Andersen, 1991).

It has previously been concluded that tectonic fracturing in the Kraka chalk may be understood as simple dome related fractures, possibly dominated by a tangential system (Jorgensen and Andersen, 1991). It was also suggested that Kraka fractures occur in swarms: a

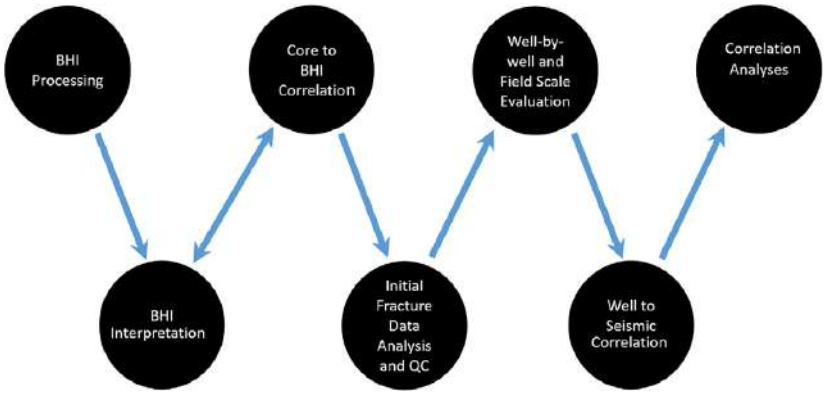


Fig. 2. Integrated workflow.

production logging tool from a horizontal wellbore indicated that only 4 of the 17 perforation intervals in that well contributed 94 %of the fluid production (Jorgensen and Andersen, 1991). The potential existence of fracture swarms is further addressed through our workflow. The results of our data analyses indicate that the Kraka fracture pattern is more complex than previously suggested with a primary set of fractures controlled by the regional maximum horizontal stress as well as a secondary set of dome-related fractures, associated with halokinesis.

3. Workflow: data availability, consistency and correlation

The natural fracture pattern of the Kraka Field was interactively characterized through borehole images (BHIs) and cores, prior to structural correlation with seismic data. The fracture characterization effort was primarily focused on the lateral fracture distribution, as the majority of Kraka wells are horizontal. Main emphasis has been put on the Ekofisk section, as it constitutes the primary target of these wellbores. Seismic was used to map faults and fracture zones away from the borehole and to identify regional structural trends.

The full integrated approach is schematically shown in Fig. 2. Specific details of each step are given in the following subsections.

3.1. BHI processing

The applied borehole image data was acquisitioned in wells drilled in the time period between 1989 and 1997. Microresistivity data in Formation MicroScanner (FMS)- or Formation MicroImager (FMI) quality was available in seven wellbores. Three wellbores surveyed by measurement while drilling technology were excluded from this study due to poor data resolution.

In the first step of the workflow, the microresistivity data were manually processed in Techlog, following the Schlumberger standard outline, illustrated in Fig. 3.

Of the wells surveyed by FMS and FMI tools, one is vertical, one is deviated at approximately 70°at reservoir level and five are horizontal (Table 1).

Compared to newer image data, the BHIs provided from the Kraka Field are of relatively poor image quality. Moreover, internal image quality variations often occur within single well sections. The latter is largely due to tool sticking, artificial signals and key seating, which is observed in most borehole images from the Kraka Field. Chalk sections directly below chert bands have been particularly hard to resolve, as the chert bands do not have planar surfaces and so cause errors in the pad alignment stage of processing. The cherts, being highly resistive compared to the chalk, are in turn well resolved. Consequently, so are chert associated fractures.

Table 1
Summary of studied BHI sections.

Well	Orientation	Tool	Length (m)
Well 1	Horizontal	FMS	691
Well 2	Deviated	FMS	730
Well 3	Horizontal	FMS	1987
Well 4	Horizontal	FMI	1704
Well 5	Horizontal	FMI	2391
Well 6	Horizontal	FMI	1681
Well 7	Vertical	FMI	205

3.2. BHI interpretation

The reservoir chalk is characterized by internal non-planar resistivity contrasts (that are not an expression of bedding features), which may confuse automatic dip-picking algorithms and lead to incorrect picks. All images have therefore been manually interpreted according to dip-picking principles for horizontal wellbores.

The more than 17 000 interpreted dip picks were subsequently subjected to structural dip removal (with respect to top reservoir). Alonghole fracture densities were calculated, and then corrected for fracture and wellbore orientation using the Terzaghi correction (Terzaghi, 1965; Peacock et al., 2003). This corrects for the sampling bias due when the fractures are near parallel to the wellbores, and allows us to compare the true density of fracture sets with different orientations. This is important in the case of Kraka since all wells are drilled from a single platform located in the centre of the anticline, and hence form a radial pattern (Fig. 4). Moreover four of the seven wells are drilled in a NW-SE direction, perpendicular to the seismically mapped faults. It is therefore essential to compare the true densities of the different fracture sets to determine the main structural controls on fracture orientation.

3.3. Core to BHI correlation

The image interpretation scheme was developed through interactive evaluation of core data in the second- and third step of the workflow. Cores were available for three of the BHI-surveyed well sections (wells 1,2 and 7). Depth matching between borehole images and cores was

Fig. 3. Processing of BHI data.

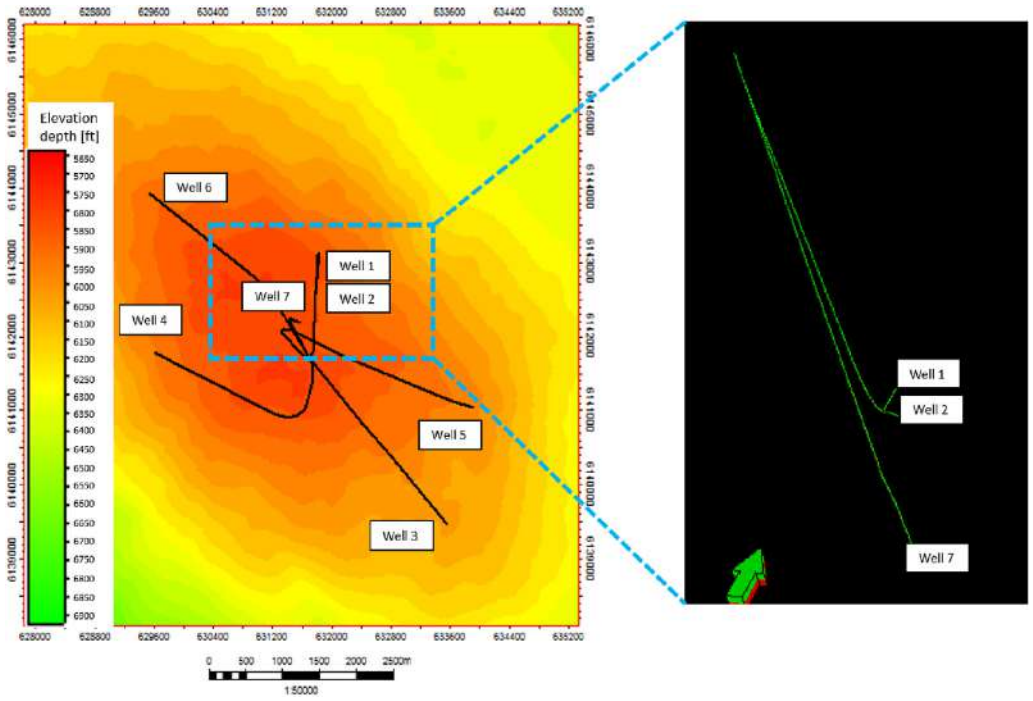


Fig. 4. Well pattern of Kraka wells included in this study on top reservoir depth map.

enabled by chert occurrences. Since the cherts are highly resistive compared to the reservoir chalk, they are easily identifiable on BHIs across the field. In the absence of a chert layer or nodule, core-to-log calibration is based on fractures. Depth shifts along wells vary by up to 8 ft. Relative fracture orientations measured in core have been reoriented, depth shifted and plotted alongside image fracture-picks in the applied software.

Correlation between borehole images and cores is considered highly advantageous because:

1. BHI and core data are complementary. Borehole images provide true orientations and survey the reservoir in-situ, so we can differentiate open and closed fractures under reservoir conditions. The advantage of core is that we can identify smaller-scale stylolite associated fractures that are not detectable on images because the image resolution is about 1–2 mm.
2. In the BHI data from the Kraka Field, sinusoids representing bedding (chalk and marl) are continuous across borehole images. Most fractures are however only represented by partial sinusoids, either because they are short or because they are only partially open or cemented. Comparison with core data, when available, is imperative in determining which partial signals should be picked. Lessons learned from cored wells are transferable to BHI-surveyed wells without core.
3. In terms of azimuth, the dip-picking tools in the applied software (as in most commercial packages) are highly sensitive to small "tweaks". This means that the orientation given by tadpoles can change drastically depending on how the partial fracture signal is picked. Where there is ambiguity, we have picked partial sinusoids to be consistent with nearby full sinusoids on borehole images, and calibrated against the fractures in core, where possible.

There is a general good correspondence between orientations of fractures identified in core and fractures picked on borehole images. An example of this is shown in Fig. 5, which shows a logged core interval (a) with corresponding BHI section (b) in the vertical wellbore.

In this case, the core contains two natural fractures, represented by blue tadpoles in the borehole image. These are recognised as natural fractures in core by the presence of slickensides. In the core section, both fractures are open. However, the logged fractures coincide with one open (conductive) and one closed (resistive) fracture picked on the image section. Here, the image interpretation is considered reliable as BHIs represent the in-situ reservoir conditions. The closed fracture observed on borehole images could possibly have been opened during the coring process itself. The dip and azimuth of both fractures identified in BHI match the orientation of the fractures logged in core within 12°.

In general, dip angles of core- and BHI fracture picks fit to within 9°or less, while dip azimuths are associated with a higher degree of uncertainty. Small discrepancies are to be expected, as core must be reoriented manually to calculate true orientations. Therefore fracture orientations from BHIs are considered the most reliable, while the presence and type of fractures can be identified in the core.

Core-fracture densities are 44% and 36% higher than BHI-fracture densities in wells 1 and 2, respectively (Fig. 6). Stylolite occurrences in the chalk accounts for some of the disparity, as they are not resolved in the images. Moreover, fractures located in bioturbated zones and well-parallel fractures are difficult to distinguish in the BHIs. The remaining discrepancy is linked to the quality and resolution of images. The fracture density percentage for well 7 has not been computed, as there are few data points to compare.

3.4. Initial fracture data analysis and QC

The true orientations of the fracture picks in each well were plotted on upper hemisphere stereonets, and these were used to classify fractures into sets on the basis of orientation. The fracture strikes were then plotted on rose diagrams drawn over the stereonets, after applying the Terzaghi correction, to enable direct comparison of the fracture densities in the different sets, on a well by well basis.

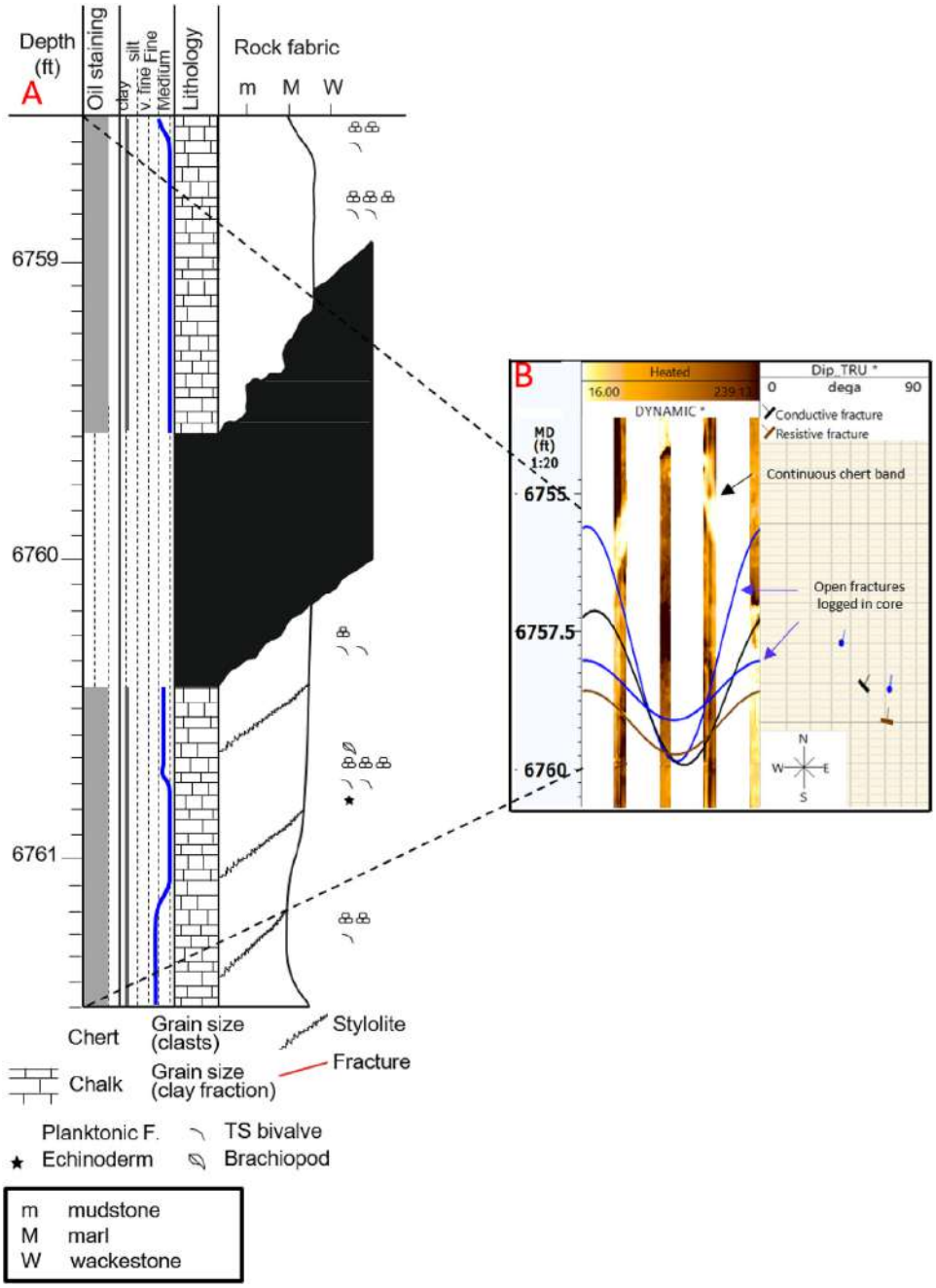


Fig. 5. Core log (a) and Core-to-BHI correlation (b) of fractured interval in the vertical wellbore. The core to log shift is approximately 2.5 ft, based on the chert band. The section of core shown corresponds to the section of borehole image log from 6756.0 to 6759.0 ft (as indicated by the black stippled lines).

3.5. Well-by-well and field scale evaluation

Alonghole fracture density logs were generated, with Terzaghi correction applied to individual fractures, to study the spatial distribution of fractures, and determine qualitatively whether they are clustered or evenly distributed. More quantitative analysis of the fracture distribution was carried out using cumulative density distribution plots, and by calculating the coefficient of variation of fracture spacing.

The cumulative fracture spacing distribution can be plotted by measuring the distance between each pair of adjacent fractures along a wellbore (without correcting for orientation), ranking them in order, and plotting the results on a cumulative density distribution diagram. The cumulative density distribution with respect to fracture spacing is analogous to the cumulative density distribution with respect to fracture displacement or fracture length (see e.g. Marrett and Allmendinger, 1991, 1992; Westaway, 1994; Marrett, 1996): a straight line on a log-linear plot indicates a random fracture distribution (Olson et al., 2001), while a straight line on a log-log plot indicates a power

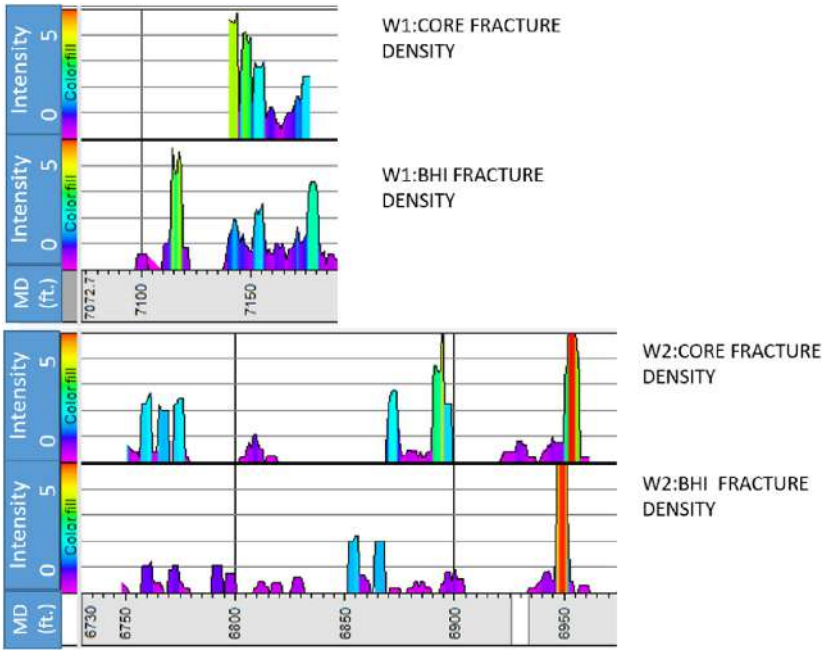


Fig. 6. Corrected fracture intensity logs of core-fractures and BHI-fractures in the cored intervals of wells 1 and 2 (scale 1:700). Fracture densities range from 0 in purple to 5 in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

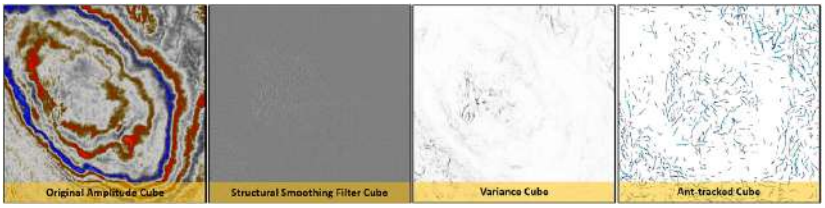


Fig. 7. Workflow for generating ant-tracked volume from the seismic depth cube of the Kraka Field in Petrel.

law distribution with clustering of fractures and may indicate fracture swarms (Gillespie et al., 1993, 2001).

The coefficient of variation of fracture spacing Cox and Lewis (1966) provides a more straightforward method to quantify the degree of clustering. The coefficient of variation R is defined as the standard deviation of fracture spacing divided by the mean fracture spacing:

$$R = \frac{\text{Standard deviation of fracture spacing}}{\text{Mean fracture spacing}} \quad (1)$$

A ratio $R > 1$ implies fracture clustering and the presence of swarms. A ratio of $R = 1$ implies a random fracture distribution A ratio of $R < 1$ implies regularly spacing fractures.

The fracture distribution can tell us something about the mechanisms of fracture formation. In particular a clustered fracture distribution often develops when the stress anomaly develops around the tip of a propagating fracture, promoting the growth of nearby fractures, in a similar manner to the process zone often observed around igneous dykes (Olson, 2003). A modelling study by Olson (2004) shows that this is often the result of critical fracture propagation in a brittle material.

3.6. Well to seismic correlation

Fracture picks from BHIs and cores were subsequently compared to a structural framework derived from the amplitude seismic volume, provided by Maersk, as well as to two ant-tracked structural models in step 6 of the workflow. The seismic amplitude cube (in depth), acquired in 2012, has a vertical resolution in the order of 40 m (sampling rate of 4 ms). The ant-tracked volumes enhance subtle faults and fracture zones that are below this vertical resolution. The ant-tracking algorithm systematically analyzes a seismic input cube – mimicking the swarm intelligence of ants (Pedersen et al., 2002). Here, a large number of agents (ants) are distributed in the volume. Each ant propagating through the cube is programmed to detect continuous structural lineations. Confidence levels are assigned depending on the length and width of the path of segments.

The first ant-tracked volume was generated in Petrel according to the following procedure (Fig. 7):

1. Cropped the original amplitude cube to speed up calculation.
2. Generated a structural smoothing/median filter cube to increase horizontal continuity and to pick out the more consistent structural features. The optimal degree of smoothing was achieved through adjusting the attribute parameter and observing its effect on the smoothing cube (in real time) prior to realization.
3. Extracted the variance/chaos cube from the smoothing cube to highlight discontinuities. The variance cube software is based on wavelet analysis. It calculates the direct measurement of dissimilarity rather than the inferred similarity of seismic data, producing sharper, more distinct results than those with traditional coherency techniques (Schlumberger, 2006).
4. Ran ant-tracking algorithm to enhance discontinuities.

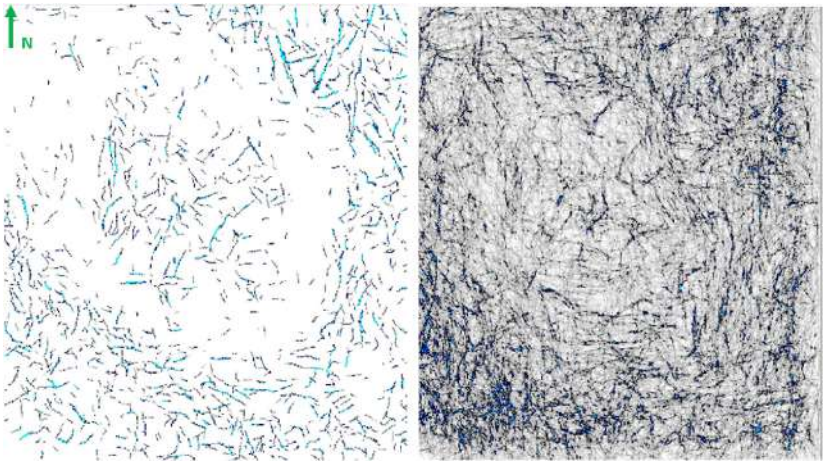


Fig. 8. Comparison of the wavelet-based (left) and the RGB-based (right) ant-tracked volumes.

The second ant-tracked volume was generated in eXchroma and Petrel according to the following procedure:

1. Cropped the original amplitude cube to speed up calculation.
2. Applied the structurally sharpened red-green-blue method, which uses the simultaneous rendering of multiple depth slices in continuous RGB color to highlight geophysical heterogeneities representative of geologic features, to the amplitude cube (Laake, 2015). The result is an image processed photo-style cube.
3. Ran ant-tracking algorithm to enhance discontinuities.

The vertical resolution of the wavelet based ant-tracked volume is 24 ms (7 times the sampling rate). In the RGB ant-tracked cube, the vertical resolution is 12 ms (3 times the sampling rate). The latter cube has therefore been preferentially used in this study (Fig. 8).

3.7. Correlation analyses

The interpretation of structural features along the well bore on seismic scale was carried out on an opacity mix of the ant-tracked volume and the seismic amplitudes (Fig. 9). This mixed view enables a focused interpretation of localized features in the seismic data. Also, we avoid misclassification of noise or acquisition artifacts. The opacity of the ant-track overlay was adjusted dynamically to enable the best interpretation possible.

The structural interpretation of the seismic data was done independently from the fracture interpretation of the BHIs. This reduces bias in the interpretation and "correlation finding" when looking at mixed displays. "Correlation finding" is a bias in interpretive science, where the interpreter has both displays open and finds feature in one display because they know to expect a feature from the other display.

Dip- and azimuth values were averaged along the fault planes of each interpreted fault to allow for direct comparison with well-scale data in upper hemisphere stereonet projections in the seventh- and final step of the workflow.

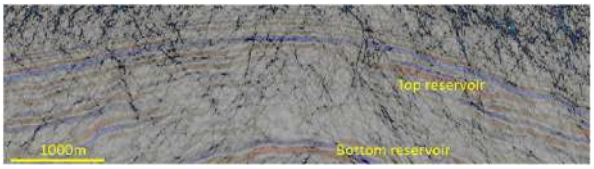


Fig. 9. Amplitude-cube over the Kraka Field with the RGB volume as opaque overlay, used for seismic interpretation.

4. Workflow outcomes: well-scale fracture trends

4.1. Fracture densities and fracture swarms

Fig. 10 shows the fracture orientations in the four NW-SE oriented horizontal wells prior to- and after applying the Terzaghi correction for borehole orientation. According to expectations, the rose diagrams for uncorrected fracture strike are dominated by NE-SW striking fractures, as fractures in this orientation will be preferentially intersected by the boreholes. After correcting for orientation, however, the rose diagrams largely remain unchanged, indicating that this reflects a real preferred fracture orientation, and not just bias due to well orientation. However in two of the wells (wells 3 and 4), the Terzaghi correction also reveals another fracture set, striking NW-SE, that cannot be identified on the uncorrected data. This suggests that multiple intersecting fracture sets are present in these locations, as well as demonstrating that the Terzaghi correction is correctly revealing the true preferred fracture orientations.

The uncorrected fracture orientations from the two wells oriented approximately north-south also show a majority of the fractures striking east-west (Fig. 11). In these wells, however, correction for wellbore orientation significantly reduces the relative importance of this fracture set, and reveals that the dominant fracture trend strikes north-south, parallel to the wellbores (although there is still a population of east-west fractures, so this location is likely to be characterized by multiple intersecting fracture sets).

In the Terzaghi-corrected fracture intensity logs from the seven wellbores, the fracture density and distribution varies between wellbores (Fig. 12). Relatively high and uniform fracture densities are observed in wells 4, 5 and 6. Mean fracture densities are lower in wells 1, 2 and 3, but the fractures in these wells appear more clustered, with potential fracture swarms observed. The highest fracture density is found in the vertical well 7 and the lowest fracture density occurs in the horizontal well 3.

The fracture distribution was investigated quantitatively by plotting cumulative density distribution plots for fracture spacing, and calculating the coefficient of variation for the fracture spacing in each well. Results were relatively consistent between all investigated wells. Generally, smaller fracture spacings (0.1–10 ft) follow a straight line on the log-log plot, indicating a close-to Power law distribution, while larger spacings (more than 10 ft) follow a straight line on the log-linear plot, indicating a close-to random distribution (Fig. 13). We can also see a clear distinction between wells 1, 2 and 3, characterized by high

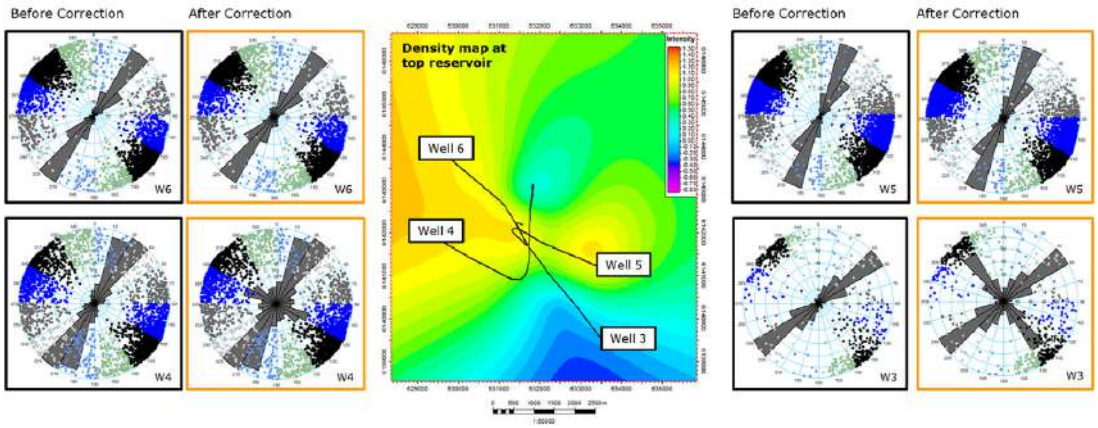


Fig. 10. Uncorrected and Terzaghi-corrected strike-rose diagrams from NW-SE oriented wells. Note that the fracture pole data, shown as dots on the stereonets (upper hemisphere), is uncorrected in both instances.

overall fracture densities (they intersect the y axis at high values) and high power law exponents (steep density distribution curves), and wells 4, 5, and 6, characterized by low overall fracture densities and low power law exponents.

The values of the mean, standard deviation and coefficient of variation R for the fracture spacings in each well are summarized in Table 2. In case of a clustered Power law distribution, the standard deviation should be smaller than the mean, so $R > 1$. In a perfectly random distribution, the standard deviation equals the mean, so $R = 1$.

This data confirms the observations from the fracture intensity plots. The mean fracture spacing is much higher in wells 1, 2 and 3 than in wells 4, 5 and 6, indicating a lower fracture density. As the coefficient of variation $R > 1$ in all wells, some clustering of the fractures occurs in all Kraka wells, but the distribution in wells 1, 2, and 3 is more clustered than in wells 4, 5 and 6. However the coefficient of variation R is still quite low in most wells (and is not much higher in wells 1 and 2 than in wells 4 and 5), suggesting a significant random component in the distribution of fractures in all wells. The greatest clustering occurs in well 3, which is the well with the lowest fracture density, i.e. the highest mean spacing.

The fracture distribution and the coefficient of variation in vertical well 7 is in good correspondence with that observed in the horizontal wells. This suggests a fairly isotropic fracture distribution in Kraka. However, this inference is based on just one horizontal well, and must therefore be treated with caution.

4.2. Fracture sets

The majority of fractures observed are steeply dipping: 0.24% of the BHI-fractures are shallow dipping (< 30°), 24.62% of the fractures have intermediate dip values (30–70°) and 75.14% are steep (70–90°) (although these figures are not corrected for orientation relative to the wellbores).

Fig. 14 shows upper hemisphere stereonets and orientation-

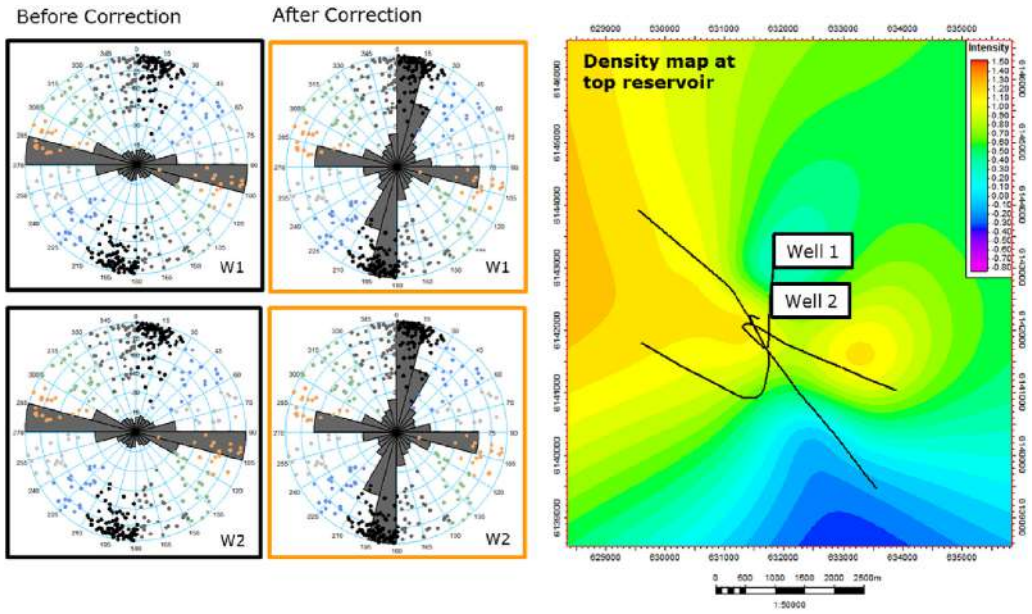


Fig. 11. Uncorrected and Terzaghi-corrected strike-rose diagrams from N–S oriented wells. Note that the fracture point data, shown in the stereonets (upper hemisphere), is uncorrected in both instances.

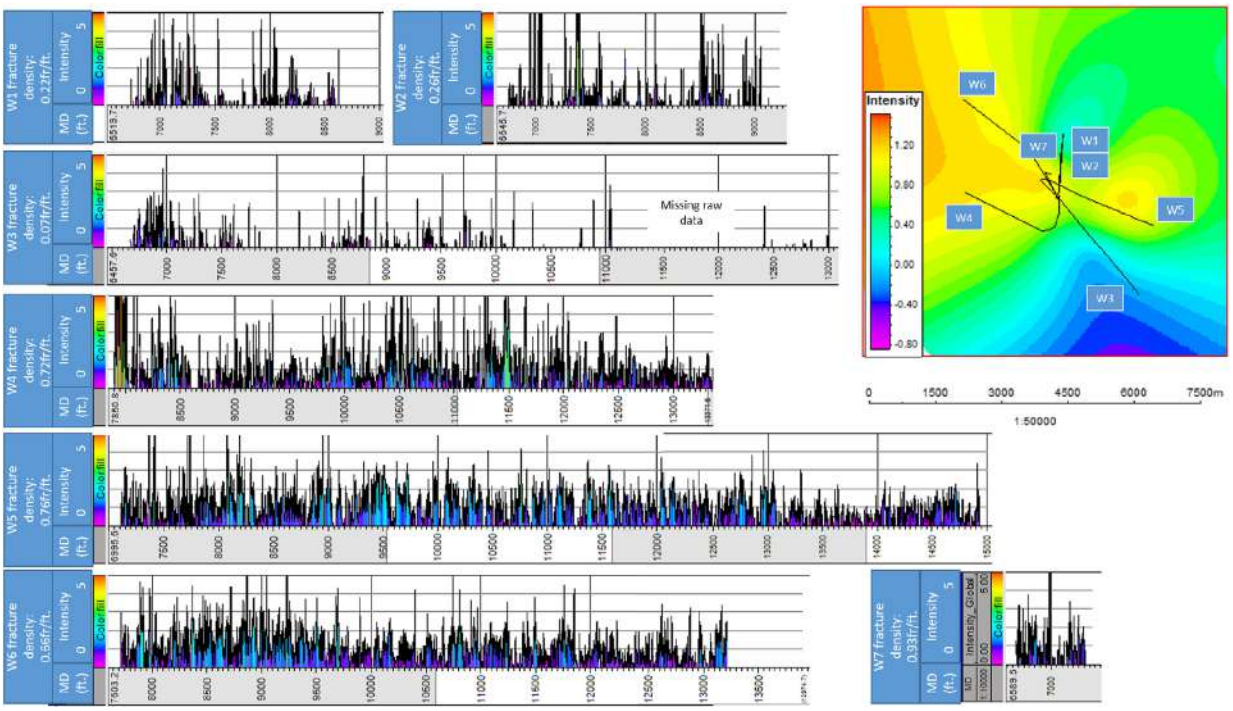


Fig. 12. Terzaghi-corrected fracture intensity logs of wells 1–7 (scale 1:10 000).

corrected rose diagrams for the fracture data from each well, broken down by stratigraphic unit. Data from the Danian Ekofisk formation is highlighted in green, and data from the Maastrichtian Tor formation is highlighted in red.

We have identified two main fracture trends in the Danian Ekofisk section. The first is a dominant NE/NNE trending regional fracture set, which strikes parallel or near-parallel to the maximum horizontal stress in the area. This main fracture trend is present in the Ekofisk intervals of all horizontal/deviated wellbores and has been confirmed by core data (from wells 1 and 2). Due to data constraints, the vertical fracture distribution of the Kraka Field was primarily studied in well 7. Although the data foundation is insufficient, results from this well indicate that the dominant NE/NNE trend of the Ekofisk formation is

Table 2
Standard deviations σ, mean values μ, and coefficient of variation R of the fracture spacing data from the seven Kraka wells.

Well	σ	μ	R
Well 1	8.3	4.2	2.0
Well 2	9.2	4.4	2.0
Well 3	21.4	7.2	3.0
Well 4	2.7	1.4	1.9
Well 5	2.2	1.2	1.9
Well 6	1.7	1.1	1.5
Well 7	4.5	2.9	1.5

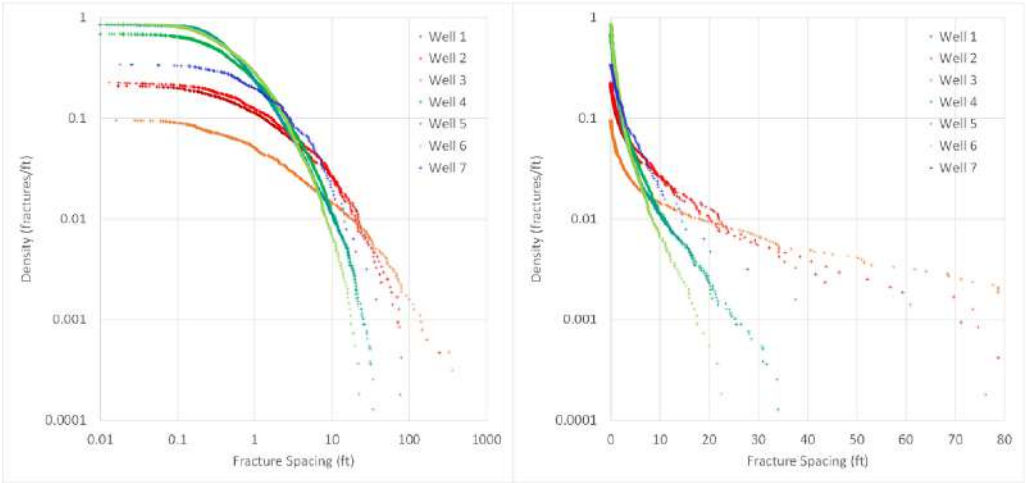


Fig. 13. Fracture spacing distributions for the seven Kraka wells, shown on a log-log plot (left) and log-linear plot (right).

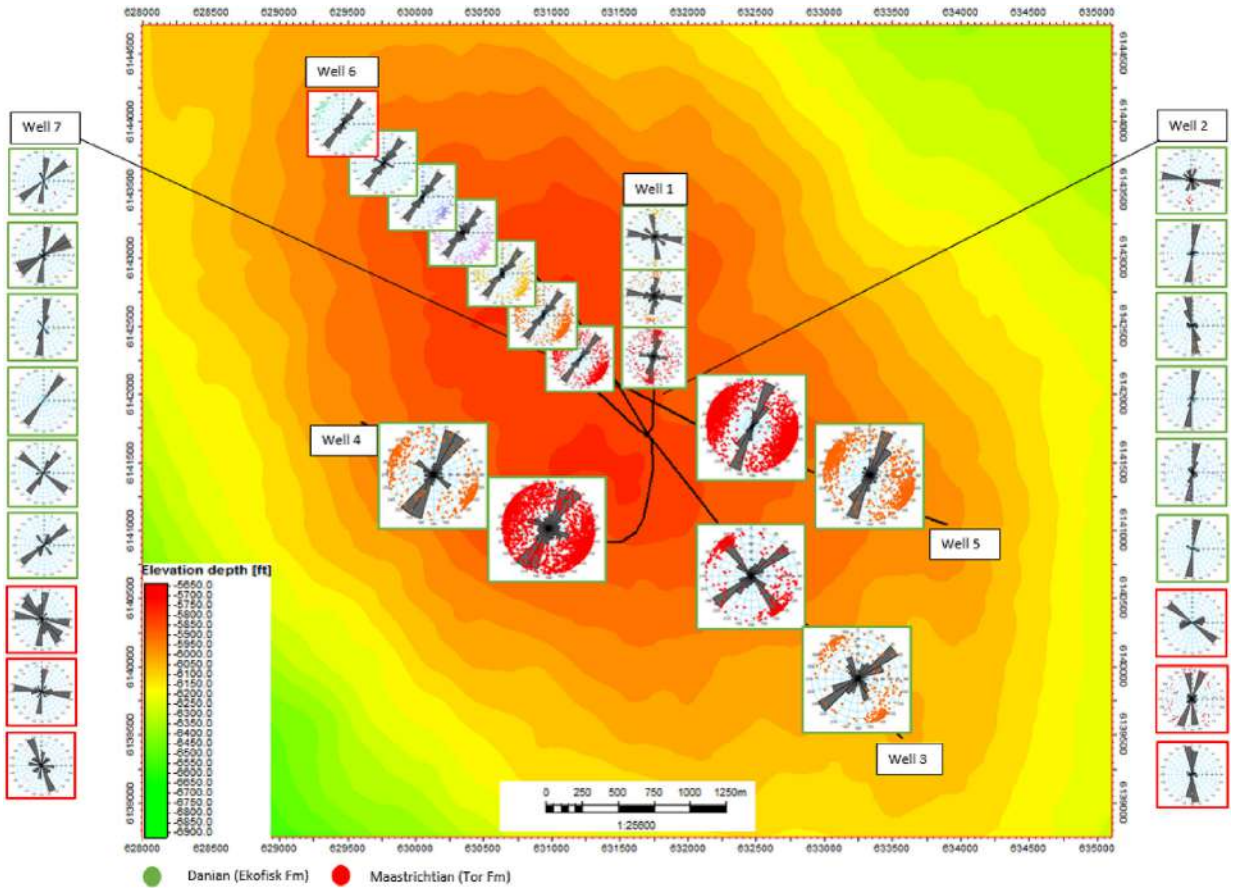


Fig. 14. Fracture point data and corrected rose diagrams in wells 1–7, plotted after unit on the top reservoir depth map.

vertically continuous.

The secondary fracture set consists of fractures striking parallel and perpendicular to the contours of the Kraka Dome. The orientation of these fractures varies between wells, depending on their location on the dome. This fracture set is thought to have formed during salt movements, and it is expected to follow the strain evolution of the Kraka chalk. Because of the positions of wells 5 and 6, the two fracture sets in cannot be distinguished on the basis of orientation in these wells (the dome contours are parallel to the NE/NNE regional trend).

The NNE/NE trending regional fracture set continues into the Tor Formation of wells 2, 6 and 7. However there is much more scatter in the Tor orientation data than in the Ekofisk data. The scattering may be due to varying local stresses during salt movements, however there is insufficient data from the Tor Formation to determine the fracture pattern with confidence.

5. Workflow outcomes: seismic-scale fault trends

The orientation data for the lineations observed on RGB ant-tracked seismic data, and for the large-scale faults interpreted on amplitude seismic are shown in Fig. 15. The ant-tracked structural model contains 25 lineations along the well trajectory. The majority (16) of these lineations strike NNE/NE. Of the 32 large-scale faults interpreted on amplitude seismic, 18 are oriented in a NE direction, while only 3 faults strike NNW. This indicates that the NNE trend is representative for smaller-scale lineations, at the limit of resolution of amplitude seismic. The rose diagram for the ant-tracked lineations shows higher variance than the large-scale faults. This implies greater variation in the orientation of small scale features in comparison to large-scale fault trends, which we would expect.

Overall the NE/NNE trend of the lineations and the large-scale faults matches the orientation of one of the main sets of fractures observed on borehole images. The main NE/NNE fracture trend can therefore be correlated from wellbore-scale fractures to local lineation scale and to field-wide fault scale (compare Figs. 14 and 15). This suggests that many of the wellbore-scale fractures are genetically related to the seismic-scale faults, and formed in response to a regional stress regime. The resulting fracture and fault set covers a range of scales, from small fractures with lengths ¡1 m up to seismic-scale faults with lengths of several km. The increased variance in orientation of the small-scale features may be partly due to uncertainty of the interpretation, and partly due to small-scale local variations in stress regime, for example around larger faults or around the salt diapir.

There are of course many more fractures observed on borehole images than lineaments observed on the RGB ant-tracked seismic data, and most of the individual fractures observed on the borehole images are of a much smaller scale than the lineaments. There is therefore no direct correspondence between fractures on borehole images and lineaments observed on the ant-tracked seismic data. Nor do we see a clear correlation between the density of fractures on borehole images and the location of the lineaments (Fig. 16). This may partly reflect variation in borehole image quality: it was not possible to pick so many fractures in areas of poor image quality.

However, if we filter the fractures from borehole images to only include the resistive fractures, we do see a clear correlation in many wells, between both the distribution and the orientation of resistive

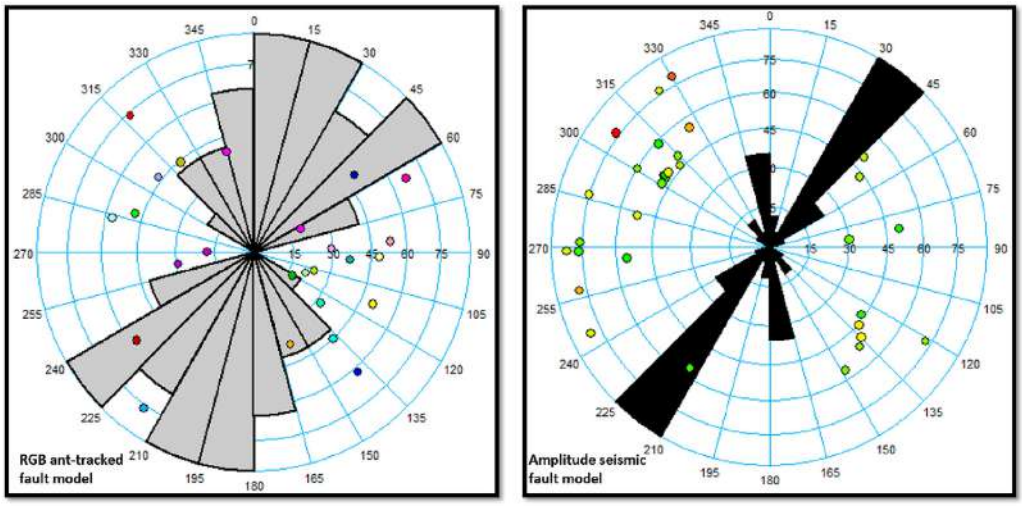


Fig. 15. Rose diagrams of structural lineations interpreted on the RGB ant-tracked volume (left) and the amplitude volume (right).

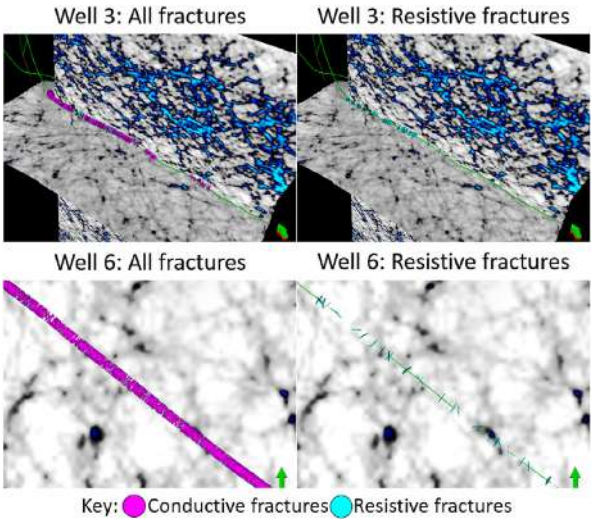


Fig. 16. Comparison of fractures interpreted on borehole images, shown by coloured disks, and lineations observed on RGB ant-tracked seismic data, for wells 3 and 6. The left hand pictures show all fractures interpreted on borehole images, with conductive fractures shown by pink disks and resistive fractures by blue disks, while the right hand pictures show only the resistive fractures. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

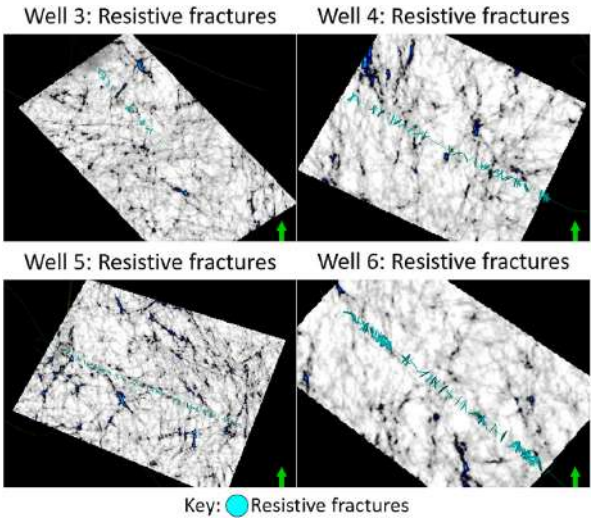


Fig. 17. Comparison of the resistive fractures interpreted on borehole images, shown by blue disks, and lineations observed on RGB ant-tracked seismic data, for wells 3, 4, 5 and 6. All wells are shown looking down from above. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

fractures and the seismic lineaments. This is particularly clear for the NW-SE oriented horizontal wells 3, 4, 5 and 6 (Fig. 17). This suggests that the resistive fractures observed on borehole images may be more likely to represent or be associated with larger-scale structures than the conductive fractures. This is quite plausible, as resistive fractures are filled with some resistive material, which may be fault gouge or cement precipitated from a fluid travelling through the fracture. Thus we would expect faults with significant displacement, or fractures associated with such small faults (e.g. in the damage zone), to appear as resistive fractures on borehole images. Conductive fractures, by contrast, could mostly represent a distributed background population of small, unfilled cracks.

6. Summary and conclusions

We have established an integrated workflow for correlation of structural features at different scales in chalk reservoirs. The workflow bridges the scale-gap between well-scale and seismic data sets, and has increased our understanding of the natural fracture system in our test case, Kraka. Such knowledge can be used strategically in optimizing production schemes and obtaining sustainable recovery factors. The combination of BHI- core- and seismic data allow for the extrapolation of fractures away from the borehole.

An obvious next step in this workflow is upscaling, fracture modelling and evaluation of current findings through fluid simulations. Such simulations, verified through production data, might provide feedback to the fracture characterization effort, further improving the model (closed loop).

Application of our approach in the Kraka Field indicate that:

1. A large portion of extensional and chert-associated fractures identified in core are distinguishable in borehole images. Stylolites, stylolite-associated fractures, fractures located in bioturbated zones and well-parallel fractures are however difficult to differentiate in the BHIs.
2. Borehole images are imperative in distinguishing cemented from open fractures, as cemented fractures may be opened during the coring process. BHIs thus increase our ability to constrain fluid flow along the fracture network.
3. Extensional and chert-associated fractures are commonly represented by partial sinusoids in BHIs, suggesting they are either short or only partially open or cemented.
4. Manual dip-picking was deemed requisite because of:
   - The large portion of fractures represented by partial sinusoids.
   - Relatively poor image quality (compared to newer BHI data).
   - Internal resistivity variations, which may “confuse” automatic dip-picking tools.
5. For the ant-tracked algorithm, higher vertical resolution can be achieved through RGB image processing, compared to wavelet based extraction of structural features. Both ant-tracked volumes display structural trends that are below the resolution of amplitude seismic.
6. Fractures picked on BHIs correlate to large-scale regional trends and to features picked on ant-tracked seismic data. This strengthens the case of a consistent regional stress field that scales down to local stresses observed at the BHI and core scale. We can therefore extrapolate fractures away from the wellbores and calibrate 3D models (e.g. discrete fracture network models).

In our test case, the results of the workflow show that the Ekofisk Formation of the Kraka Field is characterized by steep fractures striking NE and NNE, parallel or near-parallel to the maximum horizontal stress in the area. Fractures in Kraka occur as swarms and as isolated features. Moderate fracture clustering occurs in the majority of horizontal wells, as well as in the vertical well. The greatest tendency for fracture swarm occurrence is observed in the horizontal well with the lowest associated fracture density.

The main fracture trend, established from borehole images and core, is present in the Ekofisk sections of all horizontal/deviated wellbores and has been confirmed by core data. Results from the vertical well 7 indicate that the dominant NE/NNE trend of the Ekofisk Formation is vertically continuous. A secondary fracture set of fractures striking parallel and perpendicular to the contours of the Kraka Dome was identified. The orientation of these fractures varies between wells, depending on their location on the dome. This fracture set likely developed during salt movements and is expected to follow the strain evolution of the Kraka chalk.

The main NNE/NE fracture trend can be correlated from well scale to ant-tracked scale. Faults mapped on amplitude seismics can also be identified in the ant-tracked cube. In the amplitude model, faults mainly trend NE, indicating that the NNE trend is representative for smaller-scale lineations (well scale to ant-tracked scale).

This integrated study proves invaluable in testing assumptions in building fracture models and the subsequent upscaling process and will be useful for validating a geomechanically based DFN for the Kraka Field.

Acknowledgements

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre (DHRTC) under the Advanced Water Flooding programme (AWF project no: 16/00787/ginet).

We thank DUC for providing data access and F. Amour for his useful input. We would also like to extend our thanks to Schlumberger for providing licenses for Petrel, eXchroma and Techlog.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.marpetgeo.2019.104065.

References

Cox, D.R., Lewis, P.A.W., 1966. The Statistical Analysis of Series of Events. Springer Netherlands.

Fernø, M.A., 2012. Enhanced oil recovery in fractured reservoirs. In: Introduction to Enhanced Oil Recovery (EOR) Processes and Bioremediation of Oil-Contaminated Sites. InTech.

Gillespie, P., Howard, C., Walsh, J., Watterson, J., 1993. Measurement and characterisation of spatial distributions of fractures. Tectonophysics 226, 113–141. https://doi.org/10.1016/0040-1951(93)90114-y.

Gillespie, P., Walsh, J., Watterson, J., Bonson, C., Manzocchi, T., 2001. Scaling relationships of joint and vein arrays from the burren, co. clare, Ireland. J. Struct. Geol. 23, 183–201. https://doi.org/10.1016/s0191-8141(00)00090-0.

Jorgensen, L., Andersen, P., 1991. Integrated study of the kraka field. In: Offshore Europe, Society of Petroleum Engineers.

Klinkby, L., Kristensen, L., Nielsen, E.B., Zinck-Jørgensen, K., Stemmerik, L., 2005. Mapping and characterization of thin chalk reservoirs using data integration: the kraka field, Danish north sea. Pet. Geosci. 11, 113–124. https://doi.org/10.1144/1354-079304-632.

Koestler, A., Reksten, K., 1992. Insight into the 3d fracture network of an exposed analogue of fractured chalk reservoirs–the lägerdorf case. In: Proc. Fourth North Sea Chalk Symposium.

Laake, A., 2015. Structural interpretation in color — a new RGB processing application for seismic data. Interpretation 3, SC1–SC8. .

Marrett, R., 1996. Aggregate properties of fracture populations. J. Struct. Geol. 18, 169–178. https://doi.org/10.1016/s0191-8141(96)80042-3.

Marrett, R., Allmendinger, R.W., 1991. Estimates of strain due to brittle faulting: sampling of fault populations. J. Struct. Geol. 13, 735–738. https://doi.org/10.1016/0191-8141(91)90034-g.

Marrett, R., Allmendinger, R.W., 1992. Amount of extension on ”small” faults: an example from the viking graben. Geology 20, 47. https://doi.org/10.1130/0091-7613(1992)020<0047:aoeosf>2.3.co;2.

Møller, J.J., Rasmussen, E.S., 2003. Middle jurassic–early cretaceous rifting of the Danish central graben. The Jurassic of Denmark and Greenland. Geol. Surv. Den. Greenl. Bull. 1, 247–264.

Olson, J.E., 2003. Sublinear scaling of fracture aperture versus length: an exception or the rule? J. Geophys. Res.: Solid Earth 108. https://doi.org/10.1029/2001jb000419.

Olson, J.E., 2004. Predicting fracture swarms–the influence of subcritical crack growth and the crack-tip process zone on joint spacing in rock. Geol. Soc. Lond. Spec. Publ. 231, 73–88. https://doi.org/10.1144/GSL.SP.2004.231.01.05.

Olson, J.E., Yuan, Q., Holder, J., Rijken, P., 2001. Constraining the spatial distribution of fracture networks in naturally fractured reservoirs using fracture mechanics and core measurements. In: SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers.

Peacock, D., Harris, S., Mauldon, M., 2003. Use of curved scanlines and boreholes to predict fracture frequencies. J. Struct. Geol. 25, 109–119. https://doi.org/10.1016/s0191-8141(02)00016-0.

Pedersen, S.I., Randen, T., Sonneland, L., Steen, Ø., 2002. Automatic fault extraction using artificial ants. In: SEG Technical Program Expanded Abstracts 2002. Society of Exploration Geophysicists.

Rank-Friend, M., Elders, C.F., 2004. The evolution and growth of central graben salt structures, salt dome province, Danish north sea. Geol. Soc. Lond. Mem. 29, 149–164. https://doi.org/10.1144/gsl.mem.2004.029.01.15.

Rasmussen, E.S., Vejbæk, O.V., Bidstrup, T., Piasecki, S., Dybkjær, K., 2005. Late cenozoic depositional history of the Danish north sea basin: implications for the petroleum systems in the kraka, halfdan, siri and nini fields. Geol. Soc. Lond. Pet. Geol. Conf. Ser. 6, 1347–1358. https://doi.org/10.1144/0061347.

Schlumberger, 2006. Fault resolution improved for norsk hydro's varg field. case study: variance cube software enhances imaging and reduces processing time. http://www.slb.com//media/Files/software/case_studies/varcube_cs.ashx, Accessed date: 20 November 2017.

Terzaghi, R.D., 1965. Sources of error in joint surveys. Geotechnique 15, 287–304. https://doi.org/10.1680/geot.1965.15.3.287.

Westaway, R., 1994. Quantitative analysis of populations of small faults. J. Struct. Geol. 16, 1259–1273. https://doi.org/10.1016/0191-8141(94)90068-x.

4.3 Gaussian Mixture Models for Robust Unsupervised Scanning-Electron Microscopy Image Segmentation of North Sea Chalk

Abstract: Scanning-Electron images from North Sea Chalk are studied for important rock properties. To relieve this manual labor, we investigated several standard image processing methods that underperformed on complicated chalk. Due to the lack of manually labeled data, deep neural networks could not be adequately applied. Gaussian Mixture Models learnt a two-fold representation that separated the background well from the rock. Subsequent morphological filtering cleans up the prediction and enables automatic analysis.

Key points:

- Unsupervised method does not need interpreted data
- Gaussian mixture model (GMM) can distinguish chalk in Backscatter Scanning-Electron Microscopy (BSEM)
- Morphological filtering cleans up edges of chalk
- Manual task of interpretation can be automated by image analysis

J. S. Dramsch, F. Amour, and M. Lüthje (2018a). "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201803014. URL: https://doi.org/10.3997/2214-4609.201803014

Introduction

In the oil and gas industry, assessment and prediction of the hydrocarbon reserves and flow properties throughout a chalk reservoir lifetime relies, among others, on conventional and special core analysis (CCAL and SCAL) and computed tomography (CT) imaging in order to characterise the petrophysical properties and 3-D pore network geometry of chalk.

The latter laboratory experiments are technically challenging, costly, and time-consuming and require a large amount of core material. Various image analysis techniques, studying the 2-D distribution of grains, pores, and pore throats on thin-sections, have been extensively tested over more than 50yrs for workflow optimization.

Nevertheless, such techniques have not yet been integrated by reservoir engineers and geoscientists as a routine task during reservoir characterization, especially, due to a limited number of samples tested or a spatially-restricted study area that do not allow the results to be statistically representative of the chalk heterogeneity across a reservoir and between oil and gas fields.

Back-scattered electron microscopy (BSEM) analysis historically has been very manual work. Separating grains from the background, measuring perimeter and area of the grains. Recently, publications showed automatic segmentation of BSEM images using computational methods. The present study represents a robust method in the application of machine learning on thin-section images collected by BSEM. This cheap and relatively rapid technique allows to quickly analyse a large number of pictures that do not need to be manually labeled.

SEM Analysis as Image Segmentation

Scanning Electron Microscopy (SEM) is an imaging method that allows the visualisation of the grains and pores of chalk deposits (Figure 1). Grayscale images of the rock fabric can be collected at various scales of observation, from the micro-scale, typically single pore and grain, to few tens of microns where the network of pores can be studied, to the millimetres-scale. This provides a complete insight of the heterogeneity of each sample. Nanotube SEM and many applications separate very well the grains from the background in the SEM images. Therefore, these images can be segmented by histogram methods. Carbonates and specifically chalk vary on grayscale, and grains are not illuminated homogeneously. However, image segmentation has made many improvements in recent years, which extends the toolkit beyond histogram segmentation.

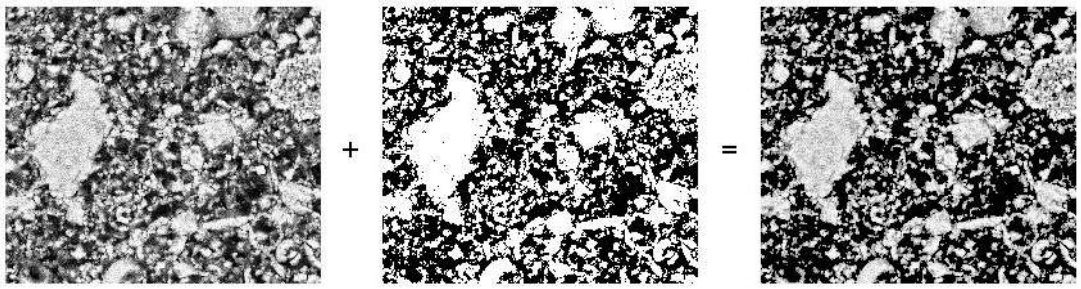


Figure 1: *Original SEM image, binary mask obtained by GMM, and resulting grain image.*

Modern Neural Networks (NN) can segment images exceptionally well (Ronneberger et al. 2015). Modarres et al. (2017) investigated the application of NNs to SEM images. However, as with most applications in Geoscience and supervised learning, we would have to label a significant amount of images by hand to assure quality or automatically with subpar methods to train the network adequately. This defeats the point for this application, therefore, this study investigates unsupervised methods, which will be assessed in order to select the one that performs the best across all scales of

observation. Several BSEM images of the rock fabric at the same scale are also collected to validate the results.

Gaussian Mixture Model (GMM) learns a number of joint distributions approximated by Gaussians in the search space (Lindsay, 1995). The number of Gaussians has to be specified, similar to many clustering methods, like k-means. In this application, we aim at segmenting the background from the chalk, which lends itself to specify two Gaussian distributions as learning parameter to obtain a binary mask, presented in Figure 1.

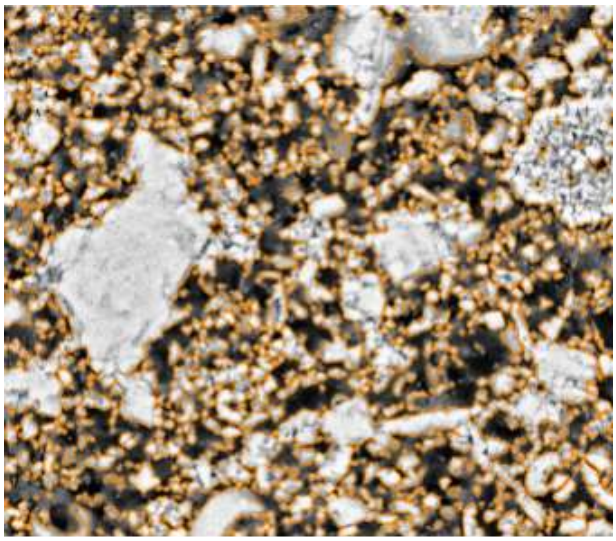


Figure 2: *Filtered segmentation of BSEM*

Morphological Filtering
We apply morphological filtering to clean up the segmentation (Serra and Vincent 1992). Due to the noisy images of BSEM, the edges of grains appear fuzzy. For the automatic analysis of the perimeter for instance, seen in Figure 2.

Subsequently, we can programmatically analyse the result using scikit-learn and scikit-image (Pedregosa 2011). This provides area, perimeter and rotation of grains in the image among other geometrical factors of the grains. These can be very valuable in digital rock physics and pore analysis.

Conclusions
We present an effective segmentation method for BSEM image data. Gaussian Mixture Models learn a good representation of the grayscale data and morphological filtering further improves the results.

Acknowledgements
The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program.

References
Lindsay, B. G. (1995, January). Mixture models: theory, geometry and applications. In *NSF-CBMS regional conference series in probability and statistics* (pp. i-163). Institute of Mathematical Statistics and the American Statistical Association.

Modarres, M. H., Aversa, R., Cozzini, S., Ciancio, R., Leto, A., & Brandino, G. P. (2017). Neural Network for Nanoscience Scanning Electron Microscope Image Recognition. *Scientific reports*, *7*(1), 13282.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, *12*(Oct), 2825-2830.

Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.

Serra, J., & Vincent, L. (1992). An overview of morphological filtering. *Circuits, Systems and Signal Processing*, *11*(1), 47-108.

CHAPTER 5

Foundations of Deep Learning for Seismic Data Analysis

This chapter is comprised of four papers that explore foundational research in Convolutional Neural Networks and signal processing.

Papers:

J. S. Dramsch, A. N. Christensen, and M. Lüthje (2019a). "Let's do the Time Warp again! – Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data". In: *Geophysics*. In Review, Chapter 5

J. S. Dramsch, M. Lüthje, and A. N. Christensen (2019g). "Complex-valued neural networks for machine learning on non-stationary physical data". In: *Computers & Geoscience*. Submitted, Chapter 5

J. S. Dramsch and M. Lüthje (2018d). "Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 5. EAGE. DOI: 10.3997/2214-4609.201803020. URL: https://doi.org/10.3997/2214-4609.201803020

J. S. Dramsch and M. Lüthje (2018c). "Deep-learning seismic facies on state-of-the-art CNN architectures". In: *SEG Technical Program Expanded Abstracts 2018*. Published, Chapter 5. Society of Exploration Geophysicists, pp. 2036–2040. DOI: 10.1190/segam2018-2996783.1. URL: https://doi.org/10.1190/segam2018-2996783.1

5.1 Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data

Abstract: This tutorial revisits Dynamic Time Warping for geoscientific applications. This algorithm can be used to match arbitrary time-series, which is applicable to 4D time shifts, seismic-well ties, well-to-well ties, and seismic pre- and post-stack migration. DTW is notorious to be computationally slow and expensive, while under-performing on seismic field data. We show that a choice of similarity measures, optimization, and constraints can both speed up calculation and significantly improve results. We show a full implementation in Python code on 4D seismic traces recorded over 10 years apart. Moreover, we explore recent developments in DTW and the significance to machine learning metrics

Key points:

- Tutorial paper shows full python code for Dynamic Time Warping
- Introduces Huber loss for geoscience problems
- Shows LB_Keogh as constraint for seismic warping
- Aligns traces well despite large discrepancy

J. S. Dramsch, A. N. Christensen, and M. Lüthje (2019a). "Let's do the Time Warp again! – Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data". In: *Geophysics*. In Review, Chapter 5

Let's do the Time Warp again! Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data

Jesper Sören Dramsch, Anders Nymark Christensen, Mikael Lüthje

ABSTRACT

This tutorial revisits Dynamic Time Warping for geoscientific applications. This algorithm can be used to match arbitrary time-series, which is applicable to 4D time shifts, seismic-well ties, well-to-well ties, and seismic pre- and post-stack migration. DTW is notorious to be computationally slow and expensive, while underperforming on seismic field data. We show that a choice of similarity measures, optimization, and constraints can both speed up calculation and significantly improve results. We show a full implementation in Python code on 4D seismic traces recorded over 10 years apart. Moreover, we explore recent developments in DTW and the significance to machine learning metrics.

INTRODUCTION

Seismic data analysis often relies on the comparison of two or more seismic traces. In quantitative 4D seismic analysis particularly, significant effort is invested in aligning traces that were acquired in the same location at different times. These comparisons tend to be more complicated than simple differencing of the traces, due to slight location, and imaging variations as well as physical changes in the subsurface.

Major problems in the alignment of traces are mismatches in trace and source location, new acquisition equipment, and changes in the subsurface. Location misalignment can stem from streamer feathering or obstructions due to newly built structures, which the acquisition vessel has to avoid. Differences in acquisition technology and azimuth play a role in misalignment. While technology often is matched as closely as possible, it tends to be a trade-off between improved imaging and matched acquisition. Changes in the subsurface stem from multiple effects that include geomechanics, pressure changes, temperature fluctuations, and fluid movement. These combined sources of misalignment cause changes in the ray path, mismatched wavelets, cycle-skipping (time-shifts by one cycle) and amplitude variations.

Conventional methods include the following methods with its most prominent properties:

- Windowed correlation – quick and reliable, given an appropriate window
- Optical Flow – Prone to amplitude changes and cycle-skipping
- Inversion-based methods – Expensive, sometimes model-dependent but reliable

Dynamic Time Warping (DTW) is a time series analysis tool that can be used for these comparisons of traces from the same location, acquired at different times. Within the methods to measure time-shifts, DTW forms its own distinct class, different from the aforementioned methods. DTW was introduced for seismic analysis in Hale (2013) in the context of 4D seismic image warping. In Luo and Hale (2014) they extended it to improve Least Squares Migration. Moreover, while we focus on 4D seismic applications, DTW can be used for seismic-well ties, post- and pre-stack seismic data analysis, well-log analysis, and time series classification.

In this tutorial paper we investigate DTW in Python, where we will use two traces from the Danish North Sea, courtesy of Total E&P Denmark. These are from the same location in a field, recorded over ten years apart. In terms of seismic acquisition this is the difference of oil-filled to solid state streamers, air gun design and differential steering of streamers.

We will focus on exploring different loss functions to align the trace data. Moreover, we investigate the use of

constraints in DTW to improve the results of the time warp results.

DATA PREPARATION

DTW does not need special preparation of the data. However, it can benefit from upsampling the traces to obtain a denser match of the warping result. This increases the computational cost of the algorithm. While the computational cost of standard DTW is $\mathcal{O}(n^2)$ it can be significantly reduced (Ratanamahatana and Keogh, 2004). Rakthanmanon et al. (2012) show a diverse suite of optimizations to search a trillion data points in under 120 seconds.

In Listing 1 we load the seismic traces into memory using segyio (Kvalsvik and Contributors, 2019), seen in Figure 1. We go on to slice the first 100 samples, upsample to 200 samples and cut away the first 25 samples due to edge effects (ringing). Just by visual inspection of the close up in Figure 2 we can see that comparing these traces will be complicated as the waveform is noticeably different and amplitudes vary strongly. These strong differences due to acquisition make it an excellent case for exploring the robustness of DTW.

```
import segyio
from segyio import tools
from scipy.signal import resample

f_old = 'new_trace.sgy'
f_new = 'old_trace.sgy'

with segyio.open(f_old, strict=False) as f:
    new_trace_ = tools.collect(f.trace[:]).T
with segyio.open(f_new, strict=False) as f:
    old_trace_ = tools.collect(f.trace[:]).T

new_trace = resample(new_trace_[:101],200)[25:]
old_trace = resample(old_trace_[:101],200)[25:]
sz = len(old_trace)
```

Listing 1: Load, slice and resample Seismic Traces. Output: Figure 1 and Figure 2

DYNAMIC TIME WARPING

In its essence, dynamic time warping is taking a similarity measure of every sample between two time series and then finding the optimal warp path to align these traces. By taking a step to the left or right through this matrix of similarity values the algorithm aligns these traces dynamically.

For the results to be sensible, several constraints are added:

- Every sample in both series must be matched.
- Each sample can have several matches
- The mappings must be monotonically increasing. I.e. we cannot have crossed matches between the two series

Dynamic time warping does not use a windowed approach. However, some constraints that can improve the results may be interpreted as a form of windowing. These windows constrain globally or locally, how far the algorithm searches for best matches in the warp path, further discussed in the section Constraints. In a geophysical sense this is limiting the allowable time shifts between traces, where a global constraint applies a uniform maximum and local constraint can vary the amount of time shifts in different sections of the subsurface.

FastDTW

Several attempts have been made to optimize the time to calculate DTW. One scheme to improve the warp speed is to start the search on a severely downsampled signal and progressively upsample the signal with iterative DTW searches. This approach has been named FastDTW. In our early experiments this approach does not perform well on seismic data, therefore, we will only mention the algorithm for completeness. In the section Constraints we show several global constraints that significantly improve the time shift results and lower the computational cost.

Soft DTW

Technically, DTW is not a metric as it does not satisfy the triangular inequality

$$D(k,l) \leq D(k,m) + D(l,m) \tag{1}$$

for $D()$ being the distance between two points, and k, l, and l being arbitrary points. Moreover, as it's a dynamic programming problem it is not differentiable, which is a desirable property in machine learning. Cuturi and Blondel (2017) redefine DTW as a differentiable loss-function. The DTW similarity measure - i.e. the length of the warped path between two time series - has been shown to outperform the euclidean distance for time-series classification, which makes it a valuable extension to the machine learning toolbox. With the following definition of the minimum

$$min^{\gamma}\{a_1, ..., a_n\} := \begin{cases} min_{i \leq n} a_i, & \text{for } \gamma = 0 \\ -\gamma log \sum_{i=1}^{n} \exp -\frac{a_i}{\gamma}, & \text{for } \gamma > 0 \end{cases}$$

where a_i are points along the path in a cost matrix, we can define the γ-soft-DTW

$$DTW_{\gamma}(x,y) := min^{\gamma}\{\langle A, \Delta(x,y)\rangle, A \in A_{n,m}\}$$

with $\Delta(x,y)$ being the distance matrix and A an alignment matrix containing the path.

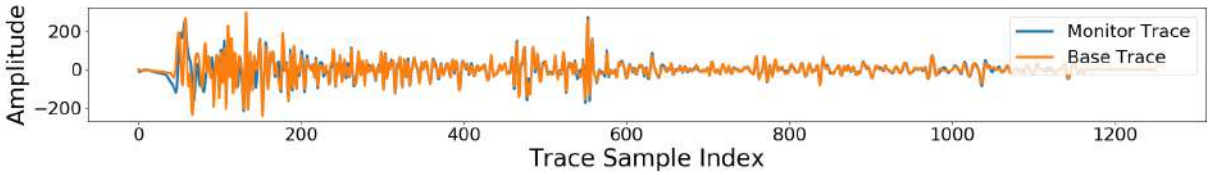


Figure 1: Comparison of Monitor and Base Trace

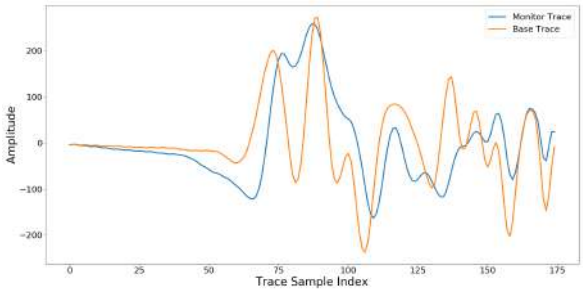


Figure 2: Close-Up of Traces in Figure 1

Dynamic Image Warping (2D/3D DTW)

Dynamic Image Warping (DIW) is the extension of DTW to 2D and 3D datasets. Hale (2013) introduced DIW for seismic data by applying the DTW algorithm in z-direction along the time-series and smoothing adjacent time-shifts to obtain a consistent results. This process can be done iteratively with progressively smaller smoothing windows to obtain x-y consistent DIW results. It is important to note that DIW does not increase the computational cost of the DTW algorithm itself. Contrary to the intuition, the distance matrixes and cumulative cost we present in the section Finding the warp, are calculated in the same way resulting in a 2D cost matrix for each pair of 1D time series. The conclusions and optimizations we present in this paper are therefore directly applicable to DIW and especially the Least-Squares Migration with DTW (Luo and Hale, 2014). Considering the speed-up to linear computational cost, DIW becomes feasible in the pre-stack domain.

DISTANCE METRICS AND SIMILARITY

A metric – or distance function – is a measure of similarity between two sets. We will consider L_1, L_2 and Huber Loss.

The warp path will be influenced by the choice of distance metric. Distances are notorious for having many different names. The easiest to calculate is the L_1 distance, also known as Manhattan distance or the (mean) absolute error.

The L_2 norm is calculated as $(a-b)^2$. It is also known as least squares or Euclidean distance.

One can see that L_1 and L_2 have some different benefitial properties and trade-offs. L_1 is linear, but non-differentiable at 0. L_2 is convex and differentiable, but the error explodes for outliers due to the square operation. Hence, the introduction of the Huber loss, which is L_2 for small values of a and L_1 for large values of a.

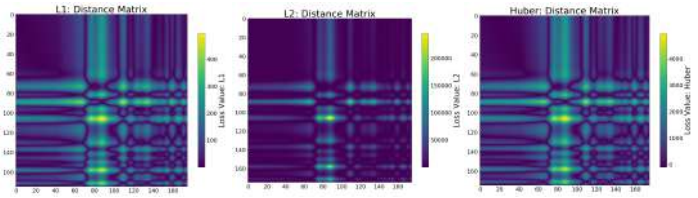
$$L_\delta(a,b) = \begin{cases} \frac{1}{2}(a-b)^2 & \text{for } |a-b| \le \delta, \\ \delta(|a-b| - \frac{1}{2}\delta), & \text{otherwise.} \end{cases} \quad (2)$$

For convenience the smooth Pseudo-Huber loss is defined as $L_\delta(a,b) = \delta^2(\sqrt{1+((a-b)/\delta)^2} - 1)$. The parameter δ is introduced to match the slope at the change from L_2 to the L_1 loss. The choice of δ can change the results for the Huber loss significantly. Experimenting on subsets of the data to obtain a good value for δ is feasible. However, looking at the variance of errors has given good results and setting δ to the standard deviation σ.

FINDING THE WARP

First, we start by building a distance or similarity matrix, with all possible combinations between the two traces, as shown in listing 2. We provide a convenient implementation of L_1, L_2, and the Pseudo-Huber loss.

In index [0,0] (using Pythonic 0-indexing) we have the distance between sample 1 in trace 1 and sample 1 in trace 2. In [2,6] we have the distance between sample 3 in trace 1 and sample 7 in trace 2, etc. Depending on the distance metric we will get different results shown in Figure 3.



(a) L_1 Distance (b) L_2 Distance (c) Huber Distance

Figure 3: Distance Matrices.

Then, we calculate the cumulative cost matrix from the distance matrix. This is achieved by first calculating the cumulative cost around the edge of the cost matrix. Then we loop over each sample of the cost matrix and fill in the values by adding the minimal value of the adjacent samples to the current sample saved in the underlying distance

matrix. This creates a cumulative cost that ideally is lowest at sample [0, 0] and increases to the last sample at [sz, sz]. The value in this last index, can beused to cheack in an optimal warp has been found. In that case the DTW similairy will match the value in [sz, sz].

We can now maximise the similarity between the traces. We search the minimum path by backtracking from [sz, sz] the bottom right corner, where we have the last points from both traces, to [0, 0] the top left corner. The minimum path will be the maximum similarity, i.e. the combination, where the traces differ the least.

An easy way to find the path is by taking the cumulative values of the similarity matrix and the use steepest descent from the lower right corner, where we simply select the direction by taking the neighbouring point with the least value. The code can be found in Appendix A and a comparison of the similarity matrix, and the obtained paths for the three different metrics can be found in Figure 5.

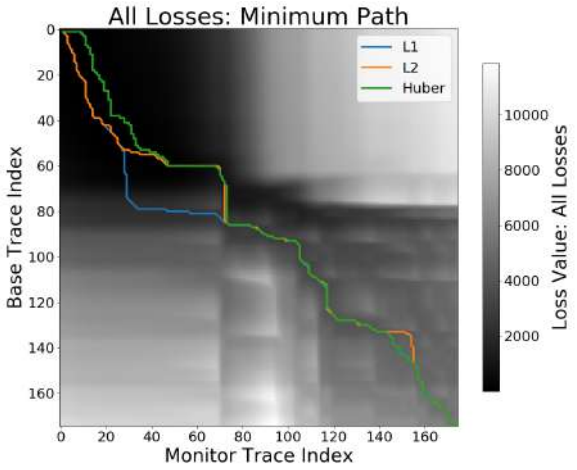


Figure 5: All Paths on Cumulative Cost.

There are several mathematical superior ways to steepest descent for finding the minimum path in a matrix. However, steepest descent is fast and easy to implement, and understand for this demonstration, as well as, many other use-cases. We show the result of the L_2 warp in Figure 7b. Mathematically, the Dijkstra algorithm is optimal for traversing a directed graph, which would be a suitable reformulation of this problem, but would exceed the scope of this tutorial (Cormen et al., 2009).

The Code Listing 2 shows the full calculation of multiple distance matrixes and calculating the cost matrix from it. On line 1 to 8 we present the three distance metrics we evaluate in this paper. On line 10 we assign the metric to a common name, therefore we don't have to rewrite the entire script. On line 12 we allocate the cost matrix with zeros in memory.

```
1  l1_distance = abs(new_trace.T-old_trace) # L1
2  l2_distance = (new_trace.T-old_trace)**2 # L2
3
4  delta = 10
5  hu_distance = ( np.sqrt( ( 1 + ( \
6                    new_trace.T  - old_trace) \
7                    / delta)**2) - 1) \
8                * delta**2 #Huber
9
10 distance = hu_distance # Choose a Distance
11
12 cost_mat = np.zeros((sz,sz))
13
14 cost_matrix[0, :] = np.cumsum(distance[0, :])
15 cost_matrix[:, 0] = np.cumsum(distance[:, 0])
16
17 for old in range(1, sz):
18     for new in range(1, sz):
19         cost_matrix[old,new] = distance[old, new] \
20             + min(cost_matrix[old-1, new-1],
21                   cost_matrix[old-1, new],
22                   cost_matrix[old, new-1])
23
24 p,q = backtrack(cost_matrix)
```

Listing 2: Calculate unconstrained cumulative cost matrix. Output: Figure 4

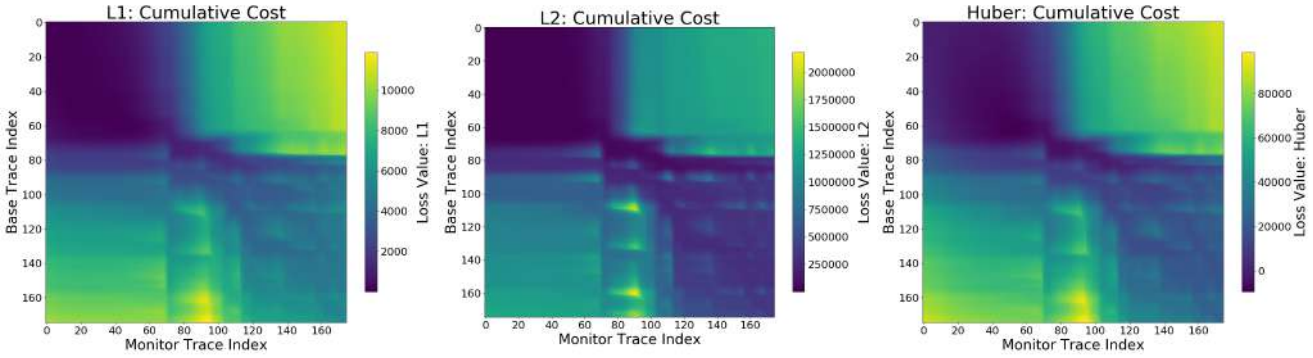
On line 14 and 15 we fill the left and upper edges of the cost matrix with the cumulative sum. Each value is the sum of the previous cumulative value and the current distance matrix. Using numpy instead of loops offloads the computation to optimized C-code in the background.

On line 17 to 22 we populate the cost matrix iteratively with the cumulative sum of the minimal value between the diagonal upper left, left and upper value in the matrix. Unfortunately, due to the nature of this dynamic programming problem, we have to implement this with two for-loops. However, significant speed-ups can be obtained by using just-in-time (JIT) compilation with Numba (Lam et al., 2015).

CONSTRAINTS

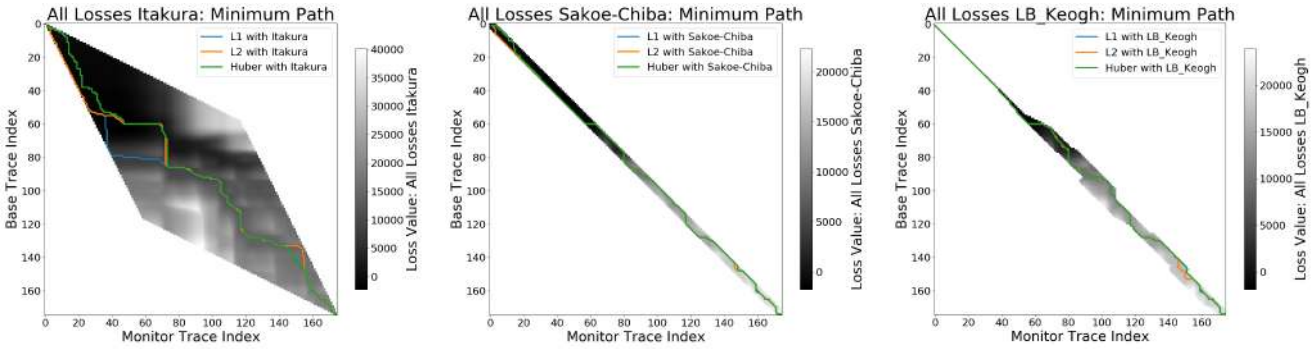
Dynamic time warping in its original form allows warping of any sample to any location within the constraints mentioned earlier. Realistically, boundary conditions improve the result, by constraining the maximum warp distance. In the physical realm of 4D this is equivalent to setting a maximum time shift. This would require some knowledge of the reservoir, but we can make some conservative estimates for 4D seismic anyways.

Several implementations were suggested to limit warping to central values of the distance matrix. Namely, the most wide-spread global constraints are the Itakura parallelogram (Itakura, 1975) and the Sakoe-Chiba disk



(a) L_1 Cost (b) L_2 Cost (c) Huber Cost

Figure 4: Cumulative Cost Matrices.



(a) Itakura (1975) Parallelogram (b) Sakoe and Chiba (1978) Disc (c) LB_Envelope (Keogh and Ratanamahatana, 2005)

Figure 6: Minimum path for constraint masks for cumulative cost in DTW.

(Sakoe and Chiba, 1978). Furthermore, we will explore the LB_Keogh lower bound for timeseries (Keogh and Ratanamahatana, 2005).

Itakura Global Constraint

The Itakura parallelogram does not take any input parameters in its original form. The general shape constrains the warp path stronger towards the ends of the time series and gives more liberties in the central region of the time series. The change of the minimum path for the three distance metrics are shown in Figure 6a.

The Code Listing 3 shows the calculation of the Itakura parallelogram constraint. On line 1 we allocate the full mask matrix with ∞ as the default value. On line 3 to 4 we define the calculate $2 * l - k$, with l and k being the indices of the matrix. On line 5 to 9 we exploit the symmetry of the parallelogram and set every sample within the parallelogram to 0.

```
1 itakura = np.full((sz,sz), np.inf)
2
3 parallel = (np.subtract.outer( \
4                range(0,sz*2,2), range(sz)))
5 itakura[(parallel   > 0)  * \
6         (parallel.T > 0)  * \
7         (parallel   < sz) * \
8         (parallel.T < sz)   \
9         ] = 0
```

Listing 3: Itakura Parallelogram Global Constraint. (Itakura, 1975) Output: Figure 6a

Sakoe-Chiba Global Constraint

The Sakoe-Chiba Disc is a uniform constraint that limits the maximum time-shifts uniformly. This leaves the path to be constrained within the center following band. E.g if we want a constraint on 5 samples, which is equivalent to

a maximum time-shift of 10 ms at a sampling interval of 2 ms, we will use a Sakoe Chiba radius of 3.

```
sa_radius = 4
sakoe = np.full((sz,sz), np.inf)
sakoe[np.abs( \
          np.subtract.outer(range(sz),range(sz)) \
      ) < sa_radius] = 0
```

Listing 4: Sakoe-Chiba Disc Global Constraint. (Sakoe and Chiba, 1978) Output: Figure 6b

The Code Listing 4 shows the calculation of the Sakoe-Chiba disc constraint. On line 1 we define the disc radius. On line 2 we allocate the full array for the mask with ∞ as the non-included cost. On line 3 to 5 we subtract the row vector of indices from the column vector of indices and apply the absolute to that difference. Then we set every sample in the mask matrix to 0, if it's within the disc radius.

Lower Bound Keogh Envelope Global Constraint

The Lower Bound Keogh constraint is based on the Ratanamahatana-Keogh constraint (Niennattrakul and Ratanamahatana, 2009), which introduced arbitrary global constraints. While these give a possibility for an application of machine learning to refine the algorithmic accuracy, we will first explore the applicability of lower bounds for time series. Rath and Manmatha (2002) prove that the LB_Keogh lower bound holds for multivariate time series.

For this to hold, $LB\_Keogh(Q,C) \leq DTW(Q,C)$ was shown to be true. The LB_Keogh lower bound is defined as follows:

$$LB\_Keogh(Q,C) = \sum_{i=1}^{n} \begin{cases} (q_i - U_i)^2, & \text{for } q_i > U_i \\ (q_i - L_i)^2, & \text{for } q_i < L_i \\ 0, & otherwise \end{cases} \tag{3}$$

with Q the Query time series with q_i being the elements of Q. C the candidate sequence, with U_i and L_i being the upper and lower envelope of C respectively. In the 4D seismic domain the query string corresponds to tho monitor trace and the candidate string corresponds to the base trace.

We calculate U and L as follows:

$$U_i = max(q_{i-r}, ..., q_{i+r}) \tag{4}$$

$$L_i = min(q_{i-r}, ..., q_{i+r}) \tag{5}$$

with r being analogous to the radius of the Sakoe-Chiba Disc, defining the perimeter for the time-series envelope.

This definition of LB_Keogh focuses on the L_2 norm. However, it can be shown that LB_Keogh holds for arbitrary distance metrics (Minkowski distances of order (p-), incuding L_1 and Huber). This allows for an efficient constraint of DTW on arbitrary metrics and arbitrary time series. Lower bounded DTW can reduce the computation from $\mathcal{O}(n^2)$, including applications to segment search Smith and Craven (2008). In geoscience this would allow for target-oriented migration results to be aligned within a larger cube as well as sequential scanning.

Listing 5 contains the envelope (U, L) of C as a hard boundary as opposed to the penalty proposed in equation 3. We combine the envelopes of the query and candidate time-series to artificially increase the search window. This is not necessary but brings possible benefits in cases of cycle-skipping.

```
keogh_radius=10
keogh = np.full((sz,sz), np.inf)

L_old, U_old = metrics.lb_envelope(old_trace, \
                        radius=keogh_radius)
L_new, U_new = metrics.lb_envelope(new_trace, \
                        radius=keogh_radius)

L = np.min((L_old, L_new), axis=0)
U = np.max((U_old, U_new), axis=0)

L /= np.max((np.abs(U),np.abs(L))) \
             / (keogh_radius)
U /= np.max((np.abs(U),np.abs(L))) \
             / (keogh_radius)

for Q in range(sz):
    for C in range(sz):
        if (L[k] < C - Q < U[k]):
            keogh[Q,C] = 0
```

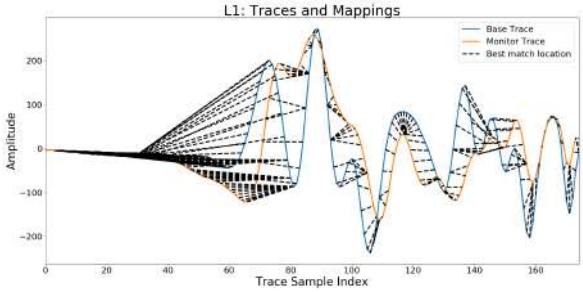
Listing 5: LB_Envelope Local Constraint. (Keogh and Ratanamahatana, 2005) Output: Figure 6c

The Code In Listing 5 we calculate the envelope-based constraint on DTW. On line 1 we define the radius for the envelope. On line 2 we allocate the mask matrix with ∞. On line 4 to 7 we calculate the envelope of both time series using the `tslearn` library, according to equation 4 and equation 5. On line 9 to 15 we combine them and scale them to the index matrix. On line 17 to 20 we iterate through both time series. If the indices are between the lower and upper bound, we set the sample in the mask matrix to 0.

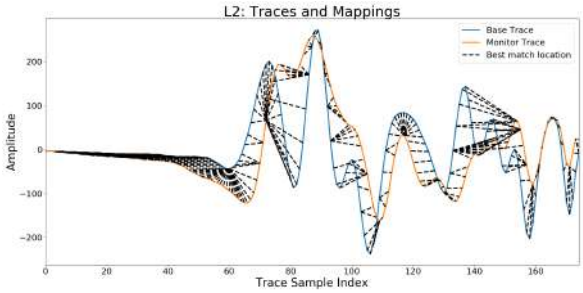
DO THE TIME WARP

The DTW algorithm may find one-to-many maps in its warp path considering that everywhere, where the path is vertical or horizontal in Figure 5 one value of the base trace is mapped to several points in the monitor trace

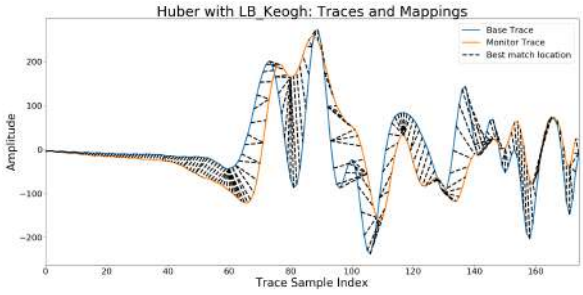
and vice versa. While technically this may be the optimal mapping between traces, this leads to non-physical interpolation results, when aligning the traces. We will handle this by taking the average value of the indices then filling the missing values. This can be achieved by using a moving average after removing duplicate values, which is shown in listing 6 and illustrated in Figure 8. By correcting this at the indexing stage, we can achieve smooth alignment of traces without deteriorating the result.



(a) Unconstrained Mapping of Traces, $L1$



(b) Unconstrained Mapping of Traces, $L2$



(c) Constrained Mapping of Traces, Huber

Figure 7: Mapping of Traces using DTW, showing benefits of constraints and distance measures.

The Code In Listing 6 we calculate the minimum path, adjust the path, and warp the monitor trace to match the base trace. On line 1 we import the 1D univariate spline interpolation. On line 3 we calculate the minimum path for the cumulative cost matrix from Listing 4 and constrain it with the keogh envelope from Listing 5. On line 5 to 7 we replace the one-to-many mappings visible in Figure 7c with the mean value. On line 9 to 12 we smooth indices to condition the indexing for the interpolation. On line 14 to 16 we interpolate the monitor trace to the smoothed indices, which results in Figure 8.

```
1  from scipy.interpolate import \
2  UnivariateSpline as interp1d
3
4  re_old, re_new = backtrack(cost_matrix + keogh)
5
6  out = [(np.mean(np.array(re_new) \
7          [np.array(re_old) == q])) \
8          for q in range(len(new_trace))]
9
10 N = 7 #Smoothing Factor
11 out_smooth = np.convolve(out,
12                           np.ones((N,))/N,
13                           mode='same')
14
15 trace_interp = interp1d(np.arange(new_trace.size),
16                          new_trace.T,
17                          k=4)
```

Listing 6: Warping of Traces constrained by LB_Keogh. Output: Figure 8

DISCUSSION

In this tutorial we revisited Dynamic Time Warping for geophysical applications. In our investigation of the method we encountered the strong dependence of DTW on the metric and constraint chosen. Moreover, we find that DTW is very difficult to use on field data without additional constraint on the warp path. The data we use in this example is field data that was recorded over 10 years apart, which introduces a multitude of complications regarding the estimation of time-shifts. We show that using an expansion of the envelope can significantly improve the warping result of seismic traces.

Distance Metrics

We find that the L_1 norm is not ideal to find correspondence in seismic traces (cf. Figure 7a). This metric does not adequately distinguish small-scale changes in the data, which is essential to obtain a good time warp result. The L_2 norm gives better results than the L_1 norm but is too prone to large scale variations in the amplitudes (cf. Figure 7b). The data we use in the example shows a change in wavelet, which makes it hard for the L_2 norm to accurately map some shifts. The Huber loss provides a good medium between the L_1 and L_2 norm. However, despite improving results, it is highly dependent on a good choice of δ, which is a drawback in comparison to the hands-off nature of the other metrics. The results of the Huber loss

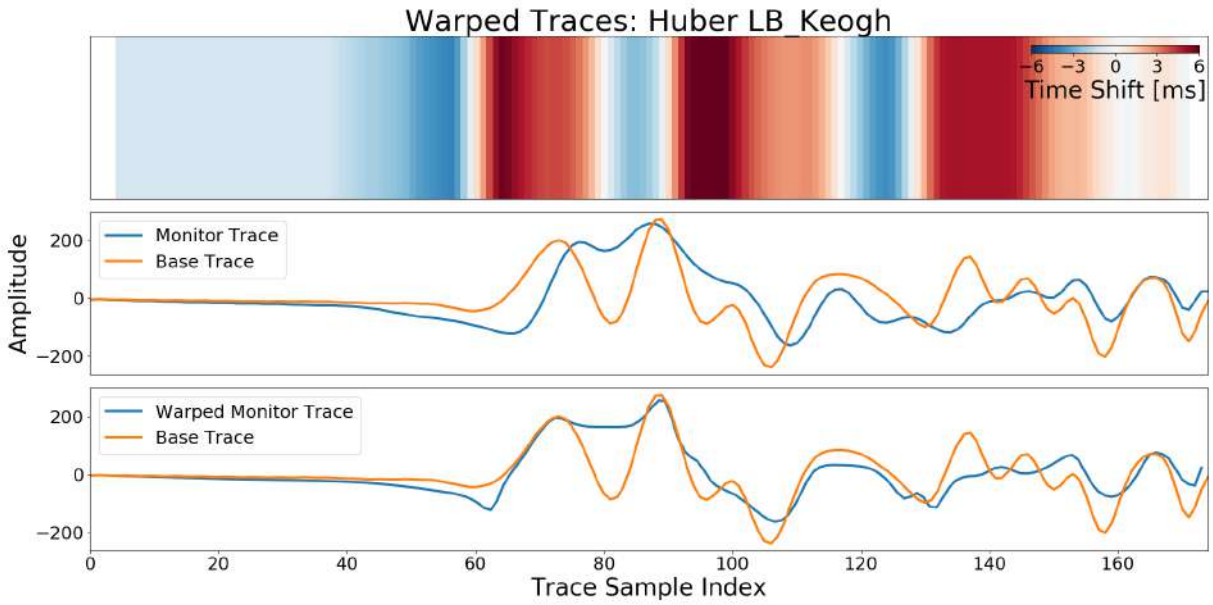


Figure 8: Time shift and warped traces for amplitude comparison using Huber loss and LB_Keogh envelope-based constraint.

itself are, however, not convincing on an unconstrained warp path. The hard nature of the particular problem we explore in this tutorial needs strong constraints to be feasible.

Constraints

Due to the particularly different nature of the problem we pose in this example, we need to better constrain the search space for the optimal warp path. The Itakura parallelogram does not perform well on the nature of geophysical problems, while the beginning of the trace varies less, the large margin for variation towards the central region of the traces is not conducive to these geophysical problems. The Sakoe-Chiba disc, which is equivalent to a global constraint of maximum time-shift gives a better overall result in the warping and time-shift result and is a good general choice for DTW problems in geophysical data. We introduce the expanded time series envelope, also used in the calculation of the Keogh lower bound, as global constraint on the search space, which provides superior results to the Sakoe-Chiba Disc. The expanded envelope allows for greater variation between regions of large amplitudes, with strong constraints on regions of low amplitudes. Due to the dynamic warping, this leads to a disentanglement of time-shifts between subsurface regions, while allowing for enough flexibility due to the expansion of the envelope to also align low amplitude regions.

Computational Cost

Implementing the constraints and subsequently calculating the cost matrix and and minimum warp path can significantly reduce the computational time. The promises of obtaining $\mathcal{O}(n)$ for DTW from other domains, are only valid for calculations of the DTW measure, they are not valid for obtaining the warp path. Nevertheless, the time cost can be reduced below $\mathcal{O}(n^2)$. These savings make DTW and its extension DIW feasible for many problems in geophysics, possibly extending to pre-stack 4D seismic warping and pre-stack migration.

CONCLUSION

In this tutorial we explore Dynamic Time Warping with reproducible code. The code presented here, is available on GitHub, where it can be copied and reproduced. However, the code examples are sufficient to run a full DTW on the data without any addition. DTW can be implemented efficiently, speeding up the calculation significantly despite being a dynamic programming problem. The constraints we introduce using expanded envelope improves the warp result significantly, but the intuitive Sakoe-Chiba Disc gives good results. These can be used to significantly speed up the calculation and reduce the problem below $\mathcal{O}(n^2)$.

Although DTW without constraints has been known to be computationally expensice and underperforming, by introducing constraints and choosing the correct metric, we can obtain convincing results even on very hard problems.

ALTERNATIVE LANGUAGES AND LIBRARIES

Other implementations in Python we recommend, are the library tslearn, pydtw, and a Soft DTW implementation. R users can refer to the excellent DTW package for R. SAS users can use this experimental implementation. The Matlab implementation is provided in the Signal Processing Toolbox.

ACKNOWLEDGMENTS

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. We thank DTU Compute for access to the GPU Cluster.

APPENDIX A
EXAMPLE BACKTRACK FUNCTION

```
def backtrack(cost_matrix):
    old = cost_matrix.shape[0] - 2
    new = old
    x, y = [old+1], [new+1]
    while (old > 0) | (new > 0):

        direction = np.argmin((
            cost_matrix[old, new],
            cost_matrix[old, new+1],
            cost_matrix[old+1, new]
        ))

        if direction == 0: ## Diagonal
            old -= 1
            new -= 1
        elif direction == 1: ## Up
            old -= 1
        elif direction == 2:  ## Left
            new -= 1
        x.append(old+1)
        y.append(new+1)
    return x[::-1], y[::-1]
```

REFERENCES

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein, 2009, Introduction to algorithms: MIT press.

Cuturi, M., and M. Blondel, 2017, Soft-dtw: a differentiable loss function for time-series: Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 894–903.

Hale, D., 2013, Dynamic warping of seismic images: GEOPHYSICS, **78**, S105–S115.

Itakura, F., 1975, Minimum Prediction Residual Principle Applied to Speech Recognition: IEEE Transactions on Acoustics, Speech, and Signal Processing, **23**, 67–72.

Keogh, E., and C. A. Ratanamahatana, 2005, Exact indexing of dynamic time warping: Knowledge and information systems, **7**, 358–386.

Kvalsvik, J., and Contributors, 2019, Segyio.

Lam, S. K., A. Pitrou, and S. Seibert, 2015, Numba: A llvm-based python jit compiler: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, ACM, 7.

Luo, S., and D. Hale, 2014, Least-squares migration in the presence of velocity errors: SEG Technical Program Expanded Abstracts 2014, Society of Exploration Geophysicists, 3980–3984.

Niennattrakul, V., and C. A. Ratanamahatana, 2009, Learning dtw global constraint for time series classification: arXiv preprint arXiv:0903.0041.

Rakthanmanon, T., B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, 2012, Searching and mining trillions of time series subsequences under dynamic time warping: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 262–270.

Ratanamahatana, C. A., and E. Keogh, 2004, Everything you know about dynamic time warping is wrong: Presented at the Third workshop on mining temporal and sequential data, Citeseer.

Rath, T. M., and R. Manmatha, 2002, Lower-bounding of dynamic time warping distances for multivariate time series: University of Massachusetts Amherst, Tech. Rep. MM-40.

Sakoe, H., and S. Chiba, 1978, Dynamic programming algorithm optimization for spoken word recognition: IEEE Transactions on Acoustics, Speech, and Signal Processing, **26**, 43–49.

Smith, A. A., and M. Craven, 2008, Fast multisegment alignments for temporal expression profiles, *in* Computational Systems Bioinformatics: (Volume 7): World Scientific, 315–326.

5.2 Complex-valued neural networks for machine learning on non-stationary physical data

Abstract: Deep learning has become an area of interest in most scientific areas, including physical sciences. Modern networks apply real-valued transformations on the data. Particularly, convolutions in convolutional neural networks discard phase information entirely. Many deterministic signals, such as seismic data or electrical signals, contain significant information in the phase of the signal. We explore complex-valued deep convolutional networks to leverage non-linear feature maps. Seismic data commonly has a lowcut filter applied, to attenuate noise from ocean waves and similar long wavelength contributions. Discarding the phase information leads to low-frequency aliasing analogous to the Nyquist-Shannon theorem for high frequencies. In non-stationary data, the phase content can stabilize training and improve the generalizability of neural networks. While it has been shown that phase content can be restored in deep neural networks, we show how including phase information in feature maps improves both training and inference from deterministic physical data. Furthermore, we show that the reduction of parameters in a complex network outperforms larger real-valued networks.

Key points:

- Smaller complex-valued Convolutional Neural Network outperforms larger real-valued Convolutional Neural Network
- Very large real-valued CNN can implicitly outlearn explicit information in smaller complex-valued CNN
- Better geoscience-related metrics necessary
- Periodic dimming effect in Frequency-Wavenumber-domain apparent

J. S. Dramsch, M. Lüthje, and A. N. Christensen (2019g). "Complex-valued neural networks for machine learning on non-stationary physical data". In: *Computers & Geoscience*. Submitted, Chapter 5

Complex-valued neural networks for machine learning on non-stationary physical data

Jesper Sören Dramsch[a,*], Mikael Lüthje[a], Anders Nymark Christensen[a]

[a] *Technical University of Denmark, Kongens Lyngby, Denmark*

Abstract

Deep learning has become an area of interest in most scientific areas, including physical sciences. Modern networks apply real-valued transformations on the data. Particularly, convolutions in convolutional neural networks discard phase information entirely. Many deterministic signals, such as seismic data or electrical signals, contain significant information in the phase of the signal. We explore complex-valued deep convolutional networks to leverage non-linear feature maps. Seismic data commonly has a lowcut filter applied, to attenuate noise from ocean waves and similar long wavelength contributions. Discarding the phase information leads to low-frequency aliasing analogous to the Nyquist-Shannon theorem for high frequencies. In non-stationary data, the phase content can stabilize training and improve the generalizability of neural networks. While it has been shown that phase content can be restored in deep neural networks, we show how including phase information in feature maps improves both training and inference from deterministic physical data. Furthermore, we show that the reduction of parameters in a complex network outperforms larger real-valued networks.

Keywords: Machine Learning, Deep Learning, Neural Networks, Physics-based machine learning, Geophysics, Seismic

1. Introduction

Seismic data is high-dimensional physical data. During acquisition, the data is collected over an area on the Earths surface. This images a 3D cube of the subsurface. Due to low reflection coefficients and low signal-to-noise ratio, the measurements are repeated, while moving over the target area. This provides a collection of illumination angles over a subsurface area. The dimensionality of this data has historically been reduced to a stacked 3D cube or 2D sections for interpreters to be able to grasp the information of the seismic data.

With the recent revolution of image classification, segmentation and object detection through deep learning (Krizhevsky et al., 2012), geophysics has regained interest in automatic seismic interpretation (classification), and analysis of seismic signals. Through transfer learning, several initial successes were presented

*Corresponding author
Email address: jesper@dramsch.net (Jesper Sören Dramsch)

in Dramsch and Lüthje (2018a). Nevertheless, seismic data has its caveats due to the complicated nature of bandwidth-limited wave-based imaging. Common problems are cycle-skipping of wavelets and nullspaces in inversion problems (Yilmaz, 2001).

Automatic seismic interpretation is complicated, as the modelling of seismic data is computationally expensive and often proprietary. Seismic field data is often not available and their interpretation is highly subjective and ground truth is not available. The lack of training data has been delaying the adoption of existing methods and hindering the development of specific geophysical deep learning methods. Incorporating domain knowledge into general deep learning models has been successful in other fields (Paganini et al., 2017).

The state-of-the-art method has been a iterative windowed Fourier transform for phase reconstruction (Griffin and Lim, 1984). Modern neural audio synthesis focuses on methods that do not require explicit reconstruction of the phase (Mehri et al., 2016; van den Oord et al., 2016, 2017; Prenger et al., 2018). Mehri et al. (2016) introduced a recurrent neural network formulation, where van den Oord et al. (2016) reformulated the synthesis network in a strided convolutional network. The original WaveNet formulation in van den Oord et al. (2016) is slow due to the autoregressive filter, warranting the parallel formulation in van den Oord et al. (2017).

We explicitly incorporate phase information in a deep convolutional neural network. These have been heavily explored in the digital signal processing community, before the recent renaissance of neural networks and deep learning. Relevant examples to seismic data processing include source separation (Scarpiniti et al., 2008), adaptive noise reduction (Suksmono and Hirose, 2002), and optical flow (Miyauchi et al., 1993) with complex-valued neural networks. Sarroff (2018) gives a comprehensive overview of applications of complex-valued neural networks in signal and image processing.

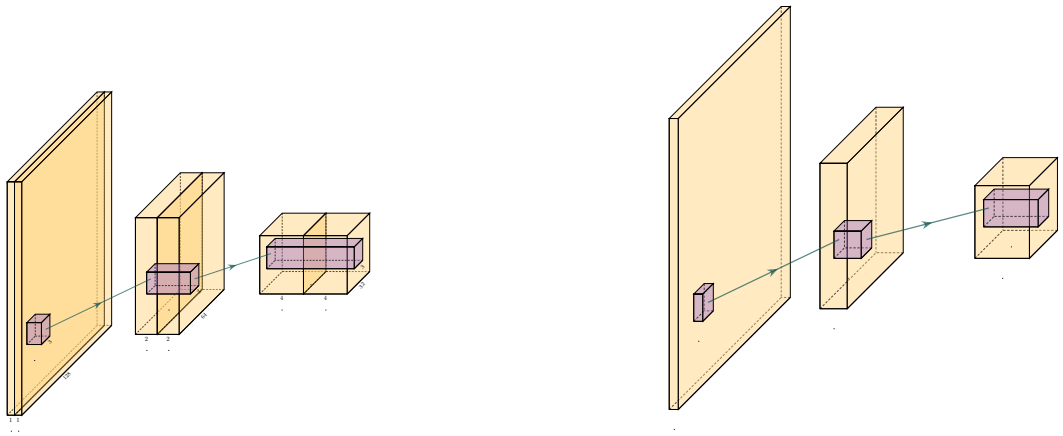
In this work, we calculate the complex-valued seismic trace by applying the Hilbert transform to each trace. Phase information has been shown to be valuable in the processing (Liner, 2002) and interpretation of seismic data (Roden and Sepúlveda, 1999; Mavko et al., 2003). Purves (2014) provides a tutorial that shows the implementation details of Hilbert transforms.

In this paper we give a brief overview of convolutional neural networks and then introduce the extension to complex neural networks and seismic data. We show that including explicit phase information provides superior results to real-valued convolutional neural networks for seismic data. Difficult areas that contain seismic discontinuities due to geologic faulting are resolved better without leakage of seismic horizons. We train and evaluate several complex-valued and real-valued auto-encoders to show and compare these properties. These results can be directly extended to automatic seismic interpretation problems.

2. Complex Convolutional Neural Networks

2.1. Basic principles

Convolutional neural networks (LeCun et al., 1999) use multiple layers of convolution and subsampling to extract relevant information from the data (see Figure 1)



(a) Complex Neural Network

(b) Real Neural Network

Figure 1: Schematic of equivalent complex- and real-valued convolutional neural network

The input image is repeatedly convolved with filters and subsampled. This creates many, but smaller and smaller images. For a classification task, the final step is then a weighting of these very small images leading to a decision about what was in the original image. The filters are learned as part of the training process by exposing the network to training images. The salient point is, that the convolution kernels are learned based on the training. If the goal is - for example - to classify geological facies, the convolutional kernels will learn to extract information from the input, that helps with that task. It is thus a very strong methodology, that can be adapted to many tasks.

2.2. Real- and Complex-valued Convolution

Convolution is an operation on two signals f and g or a signal and a filter that produce a third signal, containing information from both of the inputs. An example is the moving average filter, which smoothes the input, acting as a low-pass filter. Convolution is defined as

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau, \tag{1}$$

at the location τ. While often applied to real value signals, convolution can be used on complex signals. For the integral to exist both f and g must decay when approaching infinity. Convolution is directly generalizable to N-dimensions by multiple integrations and maintains commutativity, distributivity, and associativity. In digital signals this extends to discrete values by replacing the integration with summation.

2.3. Complex Convolutional Neural Networks

Complex convolutional networks provide the benefit of explicitly modelling the phase space of physical systems (Trabelsi et al., 2017). The complex convolution introduced in Section 2.2, can be explicitly implemented as convolutions of the real and complex components of both kernels and the data. A complex-valued data matrix in cartesian notation is defined as $\mathbf{M} = M_{\Re} + iM_{\Im}$ and equally, the complex-valued convolutional

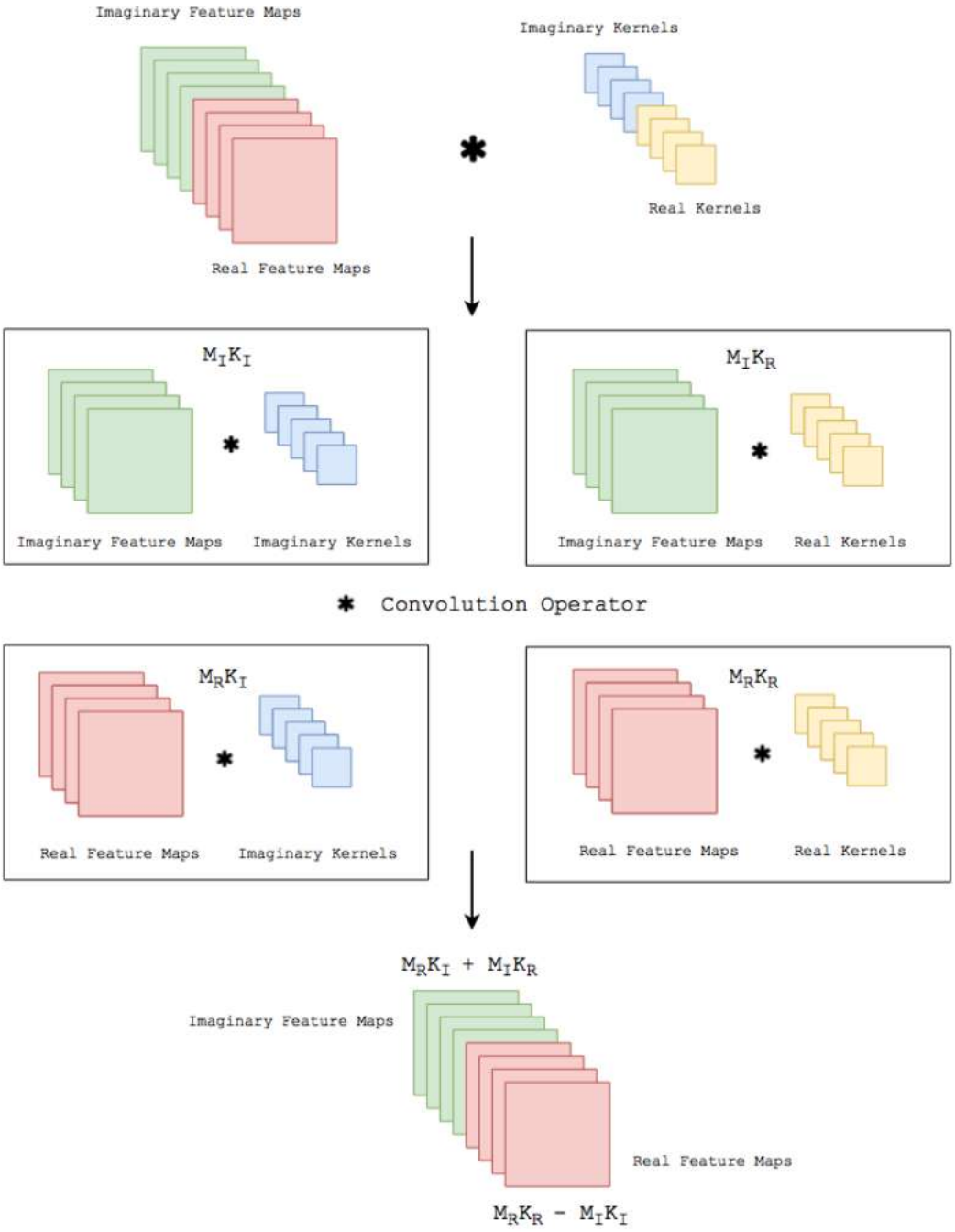


Figure 2: Implementation details of Complex Convolution CC-BY (Trabelski et al. 2017).

kernel is defined as $\mathbf{K} = K_\Re + iK_\Im$. The individual coefficients $(M_\Re, M_\Im, K_\Re, K_\Im)$ are real-valued matrices, considering vectors are special cases of matrices with one of two dimensions being one.

Solving the convolution of

$$M' = K * M = (M_\Re + iM_\Im) * (K_\Re + iK_\Im), \tag{2}$$

we can apply the distributivity of convolutions (cf. section 2.2) to obtain

$$M' = \{M_\Re * K_\Re - M_\Im * K_\Im\} + i\{M_\Re * K_\Im + M_\Im * K_\Re\}, \tag{3}$$

where K is the Kernel and M is a data vector (see Figure 2).

We can reformulate this in algebraic notation

$$\begin{bmatrix} \Re\{M * K\} \\ \Im\{M * K\} \end{bmatrix} = \begin{bmatrix} K_\Re & -K_\Im \\ K_\Im & K_\Re \end{bmatrix} * \begin{bmatrix} M_\Re \\ M_\Im \end{bmatrix} \tag{4}$$

Complex convolutional neural networks learn by back-propagation. Sarroff et al. (2015) state that the activation functions, as well as the loss function must be complex differentiable (holomorphic). Trabelsi et al. (2017) suggest that employing complex losses and activation functions is valid for speed, however, refers that Hirose and Yoshida (2012) show that complex-valued networks can be optimized individually with real-valued loss functions and contain piecewise real-valued activations. We reimplement the code Trabelsi et al. (2017) provides in keras (Chollet et al., 2015) with tensorflow (Abadi et al., 2015), which provides convenience functions implementing a multitude of real-valued loss functions and activations.

While common up- and downsampling functions like MaxPooling, UpSampling, or striding do not suffer from complex-valued neural networks, batch normalization (BN) (Ioffe and Szegedy, 2015) does. Real-valued batch normalization normalizes the data to zero mean and a standard deviation of 1. This does not guarantee normalization in complex values. Trabelsi et al. (2017) suggest implementing a 2D whitening operation as normalization in the following way.

$$\widetilde{x} = V^{-\frac{1}{2}}(x - \mathbb{E}[x]), \tag{5}$$

where x is the data and V is the 2x2 covariance matrix, with the covariance matrix being

$$V = \begin{bmatrix} V_{\Re\Re} & V_{\Re\Im} \\ V_{\Im\Re} & V_{\Im\Im} \end{bmatrix} \tag{6}$$

Effectively, this multiplies the inverse of the square root of the covariance matrix with the zero-centred data. This scales the covariance of the components instead of the variance of the data (Trabelsi et al., 2017).

2.4. Auto-encoders

Auto-encoders (Hinton and Salakhutdinov, 2006) are a special configuration of the encoder-decoder network that map data to a low-level representation and back to the original data. This low-level representation is often called bottleneck or code layer. Auto-encoder networks map $f(x) = x$, where x is the data and f

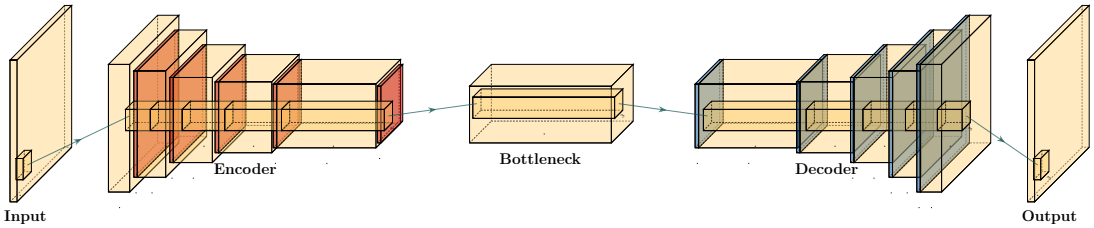


Figure 3: Typical autoencoder architecture. The data is compressed to a low dimensional bottleneck, and then reconstructed.

is an arbitrary network. The architecture of auto-encoders is an example of lossy compression and recovery from the lossy representation. Commonly, recovered data is blurred by this process.

The principle is illustrated in figure 3. The input is transformed to a low-dimensional representation - called a code or latent space - and then reconstructed again from this low dimensional representation. The intuition is, that the network has to extract the most salient parts from the data, to be able to perform a reconstruction. As opposed to other methods for dimensionality reduction - e.g. principal component analysis - an auto-encoder can find a non-linear representation of the data. The low-dimensional representation can then be used for anomaly detection, or classification.

3. Aliasing in Patch-based training

3.1. Mean-Shift in Neural Networks

A single neuron in a neural network can be described by $\sigma(w \cdot x + b)$, where w is the network weights, x is the input data, b is the network bias, and σ is a non-linear activation function. During training, the network weights w and biases b are are adjusted to a value that represents the training minimum. Learning on a mean-shift of q of an arbitrary distribution over x leads to $\sigma(w \cdot (x + q) + b)$, which increases the neuron response by q, weighted by w. During inference, both w and b are fixed, by extension the mean-shift q is fixed as well. The mean-shift over larger inference data disappears, introducing an additional bias of $w \cdot q$ before non-linear activation. This training bias may lead to prediction errors of the neuron and consequently the full neural network.

3.2. Windowed Aliasing

Non-stationary data such as seismic data can contain sections within the data that contain spurious offsets from the mean. Figure 4 shows varying sizes of cutouts, with 101 and 256 samples respectively. In the middle, the full normalised amplitude spectra are presented. On the right, the corresponding phase spectra are presented. On the left, we focus on the frequency content of the amplitude spectra around 0 Hz. The cutouts were Hanning tapered, however, a mean shift appears with decreasing patch size.

These concepts of mean-shift corresponds to a DC offset in spectral data, which can be audio, seismic or electrical data. In images this corresponds to a non-zero alpha channel. While batch normalization can

105 correct the mean shift in individual mini-batches (Ioffe and Szegedy, 2015), this may shift the entire spectrum by the aliased offset. Additionally, batch normalization may not be feasible in some physical applications pertaining to regression tasks.

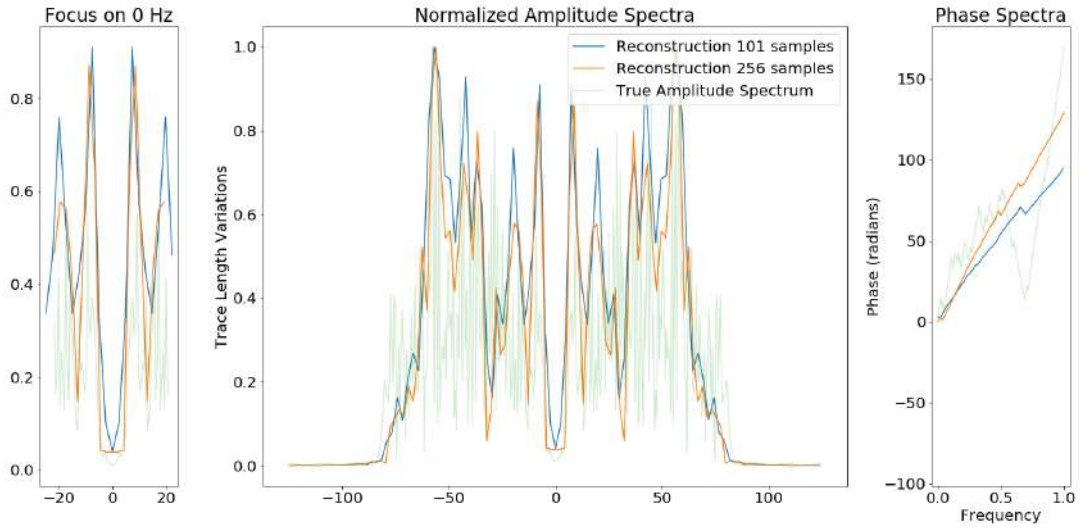


Figure 4: Spectral aliasing dependent on window-size (from Dramsch and Lüthje (2018b))

4. Complex Seismic Data

Complex seismic traces are calculated by applying the Hilbert transform to the real-valued signal. The 110 Hilbert transform applies a convolution with to the signal, which is equivalent to a -90-degree phase rotation. It is essential that the signal does not contain a DC component, as this would not have a phase rotation.

The Hilbert transform is defined as

$$H(u)(t) = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t - \tau} \, d\tau, \tag{7}$$

of a real-valued time series $u(t)$, where the improper integral has to be interpreted as the Cauchy principal value. In the Fourier domain, the Hilbert transform has a convenient formulation, where frequencies are set zero and the remaining frequencies are multiplied by 2. This can be written as

$$x_a = F^{-1}(F(x)2U) = x + iy \tag{8}$$

115 where x_a is the analytical signal, x is the real signal, F is the Fourier transform, and U is the step function. The imaginary component y is simultaneously the quadrature of the real-valued trace. This provides locality to explicit phase information, where the Fourier transform itself does not lend itself to the resolution of the phase in the time domain. In conventional seismic trace analysis, the complex data is used

to calculate the instantaneous amplitude and instantaneous frequency. These are beneficial seismic attributes for interpretation (Barnes, 2007).

5. Experiments

5.1. Data

The data is the F3 seismic data, interpreted by Alaudah et al. (2019). They provide a seismic benchmark for machine learning with accessible NumPy format. The interpretation (labels) of the seismic data is relatively coarse compared to conventional seismic interpretation, but the accessibility and pre-defined test case are compelling.

The F3 data set was acquired in the Dutch North Sea in 1987 over an area of 375.31 km^2. The sampling-rates are 4 ms in time and inline/crossline bins of 25 m. The extent being 650 inline traces and 950 crossline traces with a total length of 1.848 s. The data contains faulted reflector packets, of which the lowest one overlays a salt diapir. The data contains some noise that masks lower-amplitude events.

We generate 64x64 2D patches in the inline and crossline direction from the 3D volume to train our network. The fully convolutional architecture can predict on arbitrary sizes after training. The seismic data is normalized to values in the range of [-1, 1]. To obtain complex-valued seismic data we apply a Hilbert transform to every trace of the data and subtract the real-valued seismic from the real component (Taner et al., 1979).

5.2. Architecture

The Auto-encoder architecture uses 2D convolutions with 3x3 kernels. We employ batch normalization to regularize the training and speed up training (Ioffe and Szegedy, 2015). The down and up sampling is achieved by MaxPooling and the UpSampling operation, respectively. We reduce a 64x64 input 4 times by a factor of two to encode a 4x4 encoding layer. The architecture for the complex convolutional network is identical, except for replacing the real-valued 2D convolutions with complex-valued convolutions. The layers used are shown below (see Table 1).

Complex-valued neural networks contain two feature maps for every feature map contained in a real-valued network. Matching real-valued and complex-valued neural networks is quite complicated, as the same filter values yield a vastly different amount of parameters, as can be seen in Table 1. The smaller real-valued network contains as many feature maps for the real-valued seismic as the large complex network, the large complex network contains an additional feature map for every real-valued input for the complex component. We define a complex-valued network that effectively has the same number of filters as the real-valued small network. This network effectively has half the available feature maps for the real-valued seismic input. Moreover, we define a large real-valued network to match the number of filters of the large complex-valued network, this network has twice the feature-maps available for representation of the real-valued seismic data, compared to the large complex-valued network. The parameters are counted on the computational graph compiled by Tensorflow.

Layer (Size)	Spatial X	Spatial Y	ℂomplex Small	ℝeal Small	ℂomplex Large	ℝeal Large
Input	64	64	2	1	2	1
(ℂ)-Conv2D	64	64	8	8	16	16
(ℂ)-Conv2D + BN	64	64	8	8	16	16
Pool + (ℂ)-Conv2D + BN	32	32	16	16	32	32
Pool + (ℂ)-Conv2D + BN	16	16	32	32	64	64
Pool + (ℂ)-Conv2D + BN	8	8	64	64	128	128
Pool + (ℂ)-Conv2D	4	4	128	128	256	256
Up + (ℂ)-Conv2D + BN	8	8	64	64	128	128
Up + (ℂ)-Conv2D + BN	16	16	32	32	64	64
Up + (ℂ)-Conv2D + BN	32	32	16	16	32	32
Up + (ℂ)-Conv2D	64	64	8	8	16	16
(ℂ)-Conv2D + BN	64	64	8	8	16	16
(ℂ)-Conv2D	64	64	2	1	2	1
Parameters on Graph			100,226	198,001	397,442	790,945
Size on Disk [MB]			1.4	2.5	4.8	9.2

Table 1: Layers used in the four auto-encoders and according parameter count on the computational graph and size on disc. Complex-valued convolutions and real-valued convolutions used respectively.

5.3. Training

We train the networks with an Adam optimizer (Kingma and Ba, 2014) and a learning rate of 10^{-3} without decay, for 100 epochs. The loss function is mean squared error, as the seismic data contains values in the range of [-1,1]. All networks reach stable convergence without overfitting, shown in Figure 5.

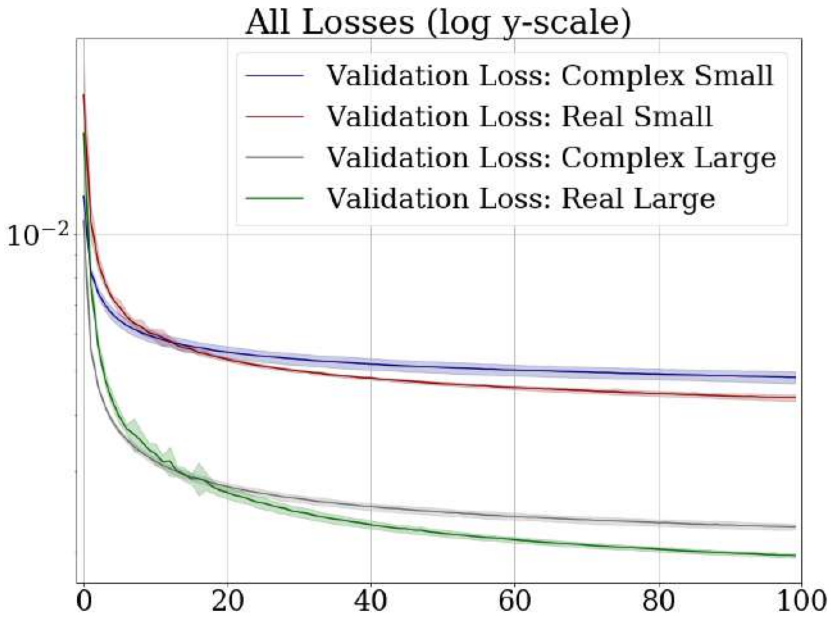


Figure 5: Validation Loss (MSE) on 7 random seeds per network. (Real-valued loss on real-valued seismic and combined complex-valued loss on complex-valued seismic, as the network "sees" it.)

5.4. Evaluation

We compare the complex auto-encoders with the real-valued auto-encoders, through the reconstruction error on unseen test data on 7 individual realizations of the respective four networks and qualitative analysis of reconstructed images. We focus on evaluating the real-valued reconstruction of the seismic data.

6. Results

We trained four neural network auto-encoders with seven random initializations for each network, to allow for error bars on the estimates in Figure 5. The mean squared error and the mean absolute error for each parameter configuration during training is given in Table 2. There is a clear correspondence of the reconstruction error of the auto-encoder to the size of network. The real-valued networks outperform the complex-valued networks in both the mean squared error and mean absolute error, however, we see that a real-valued network needs around twice as many parameters as a complex-valued network to attain the same reconstruction errors.

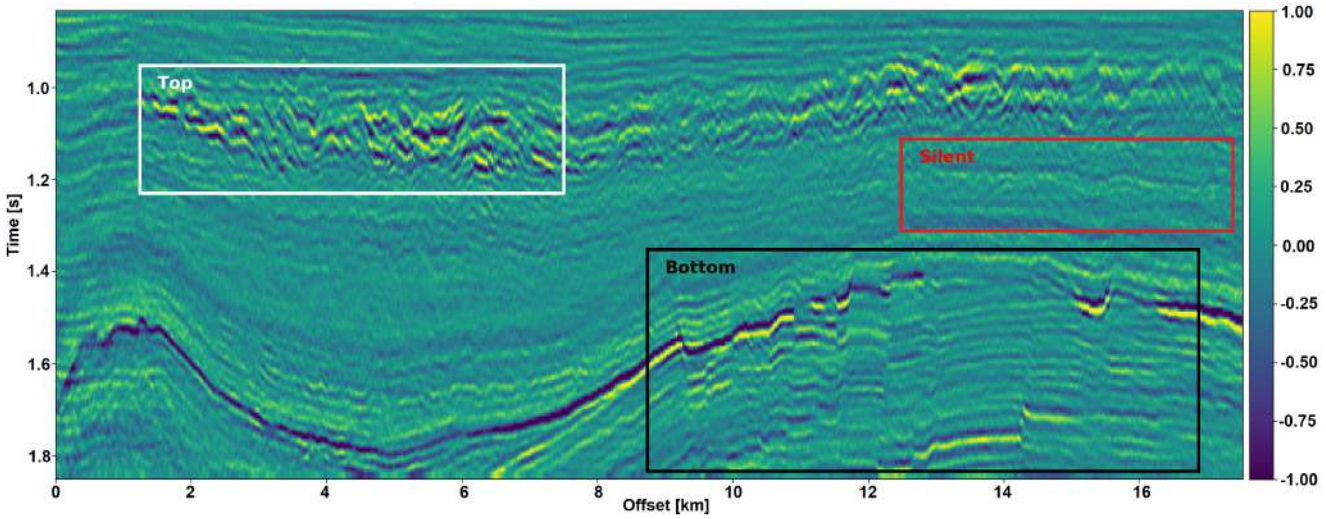
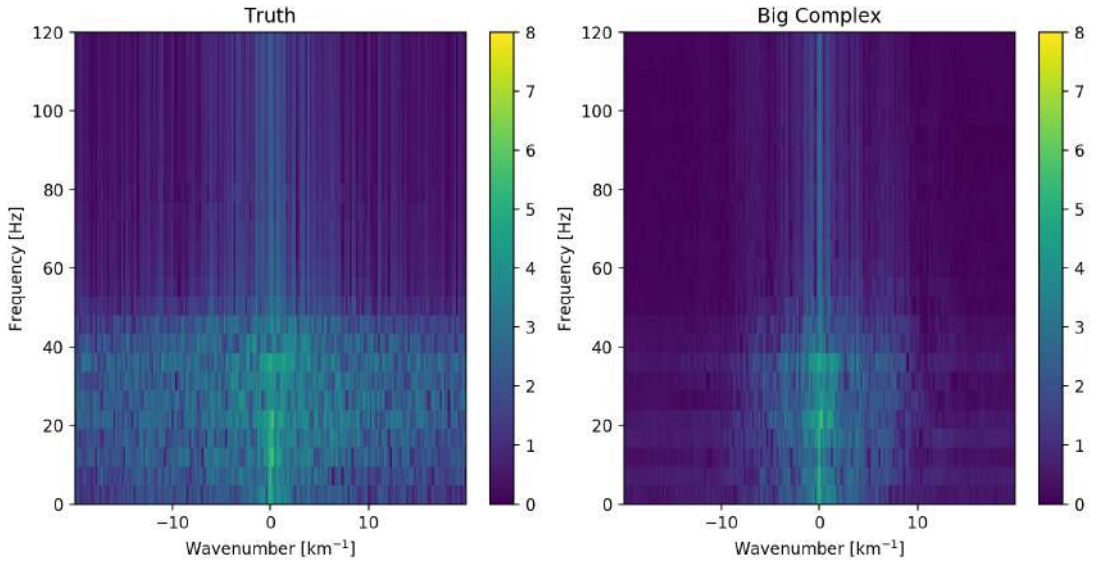


Figure 6: Seismic Test Data with marked section for closer inspection. We chose the "top" section for it's faulted chaotic texture, "bottom" for the faulted blocks, and "silent" for a noisy but geologically uninteresting section.

Network	Runs × epochs	Parameters	MSE [$\times 10^{-2}$]	MAE [$\times 10^{-2}$]
1) $\mathbb{C}_{\text{small}}$	7×100	100,226	0.484 ± 0.013	4.695 ± 0.058
2) $\mathbb{R}_{\text{small}}$	7×100	198,001	0.436 ± 0.006	4.500 ± 0.028
3) $\mathbb{C}_{\text{large}}$	7×100	397,442	0.227 ± 0.003	3.247 ± 0.025
4) $\mathbb{R}_{\text{large}}$	7×100	790,945	0.196 ± 0.002	3.050 ± 0.013

Table 2: Parameters and errors for networks (lower is better). Losses on network validation.

The seismic sections in Figure 6 show the unseen test seismic section. We perform a closer inspection of the regions "top" and "bottom" to focus on geologically relevant sections in the reconstruction process. The noisy segment without strong reflectors is a good baseline to evaluate the noise reduction of the Autoencoder and the behaviour of the different networks on low amplitude data. Overall, all networks denoise the original seismic, with the lowest reconstruction errors being RMS of 0.1187 and MAE of 0.0947 (cf. Table 3). Figure 7 shows the frequency-wavenumber (FK) of the ground truth (7 (a)) and the large complex network reconstruction (7 (b)). These show a decrease in the 0 - 60 Hz band for larger absolute wavenumbers.



(a) Ground Truth

(b) Large Complex Network

Figure 7: Evaluation on Silent Noise Patch in FK Domain. Noise reduction of frequencies below 50 Hz apparent, while reconstruction does not introduce visible aliasing.

	Full		Silent		Top		Bottom	
Network	RMS	MAE	RMS	MAE	RMS	MAE	RMS	MAE
1) $\mathbb{C}_{\text{small}}$	0.1549	0.1145	0.1265	0.1010	0.2315	0.1759	0.1588	0.1200
2) $\mathbb{R}_{\text{small}}$	0.1581	0.1153	0.1247	0.0994	0.2395	0.1810	0.1612	0.1205
3) $\mathbb{C}_{\text{large}}$	0.1508	0.1101	0.1187	0.0947	0.2301	0.1747	0.1514	0.1135
4) $\mathbb{R}_{\text{large}}$	0.1469	0.1072	0.1214	0.0967	0.2222	0.1679	0.1459	0.1088

Table 3: RMS and MAE on real component of Data Patches.

6.1. "Top" seismic section

The "top" segment contains strong reflections that are very faulted with strong reflectors. Figure 8 shows the top segment and the reconstructions of the four networks. All networks display various amounts of smoothing. The quantitative results show that the complex networks perform very similar regardless of size. The large real-valued network outperforms the complex networks by 2.5 % on RMS, while the small real-valued network underperforms by 2.5 % on RMS. The panel in Figure 8c shows a very smooth result. Despite the close score of the complex networks, it appears that the complex-valued network restores more high-frequency content. We can also see less smearing of discontinuities in the larger complex network, particularly visible in the lower part (1.2 s) at 6000 m offset, which is smeared to appear like a diffraction

in the smaller network. The large real-valued network shows good reconstruction with minor smearing with higher amplitude fidelity in areas like 1.1 s at 2000 m, however, some of the steeply dipping artifacts are visible below the reflector packet between 0 m and 2000 m offset.

6.2. "Bottom" seismic section

The data marke as "bottom" in Figure 6 contains a faulted anticline and relatively strong noise levels. The small complex network in Figure 9b reconstructs a denoised image with good reconstruction of the visible discontinuities. Some leakage of the reflector starting at 1.5 s across discontinuities is visible. The real small network in Figure 9c reconstructs a strongly smoothed image, with some ringing below the main reflector, which is not visible in the other reconstructions. The dipping reflector at an offset of 16000 m is well reconstructed, however, it seems like the reconstruction introduced ringing noise over the vertical image. The large real-valued network in Figure 9e performs best quantitatively (cf. Table 3). The complex-valued large network in Figure 9d does a fairly good job at reconstructing the image, similar to the large real-valued network. However, the amplitude reconstruction of high-amplitude events particularly in the main reflector around 1.5 s is showing.

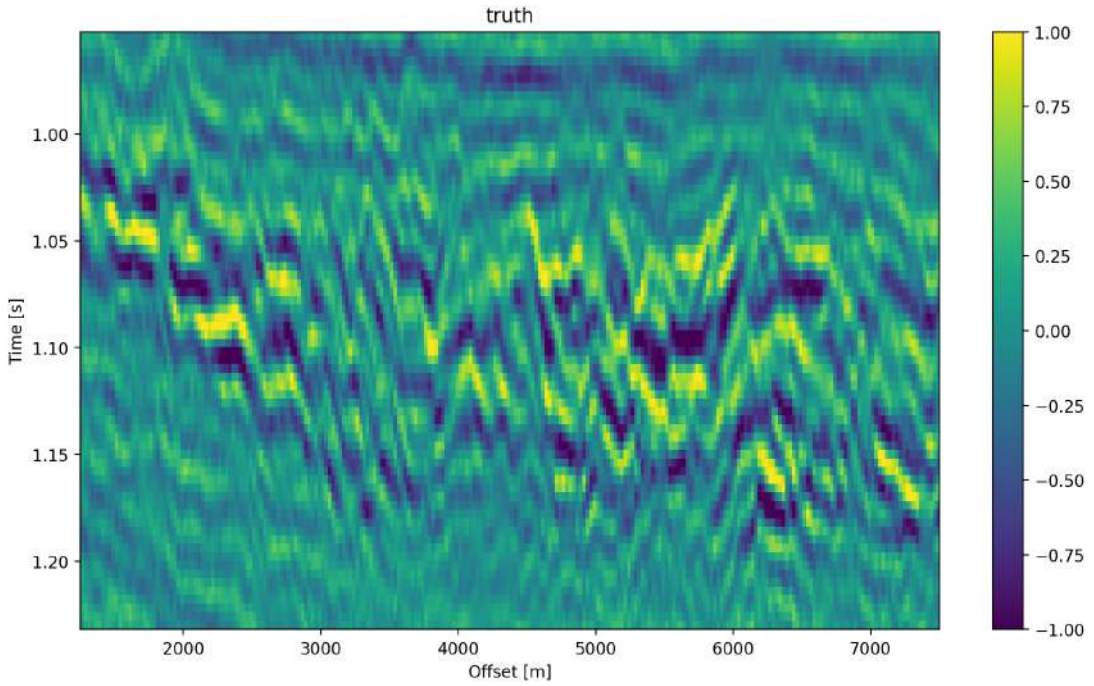
6.3. Full seismic test data

It is evident, that the small real-valued network does not match the performance of the smaller complex-valued network, even less so when compared to the large complex-valued network. We therefore compare the large networks on the full seismic data.

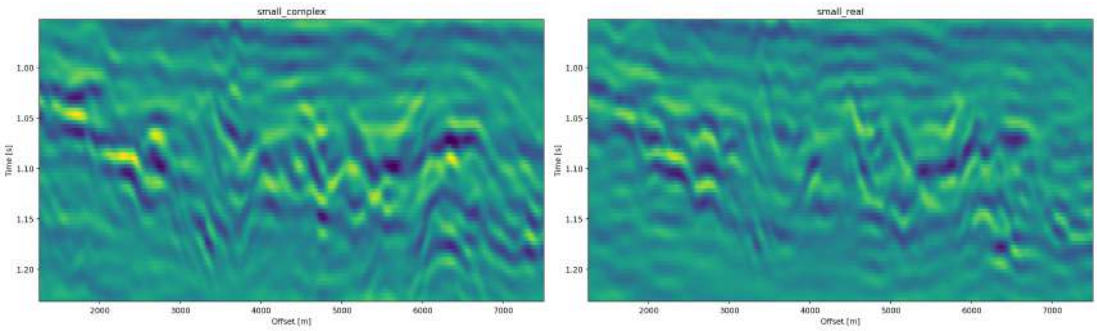
Overall, both networks return a smoothed image. The findings for the strongly faulted sections in the "top" panel hold across the entire faulted area around 1.1 s in Figure 11. The complex-valued network does a better job at reconstructing faults and discontinuities. The real-valued network is better at reconstructing high-amplitude regions that appear dimmer in the complex-valued region. The reconstruction of both networks seems adequately close to the ground truth, with differences in the details. Quantitatively, the real-valued network does the better reconstruction in Table 3 with an improvement of 2.5 % over the large complex-valued network. The FK domain shows a very similar reduction in noise in the sub 50 Hz band in Figure 10. All networks introduce an increase of energy across all frequencies at wave-number $k = 0\ km^{-1}$. Additionally, a dimming of the frequencies around $k = 2.5\ km^{-1}$ appears in all reconstructions, but is more prominent in the large complex-valued network. The ground truth seismic contains some scattered energy in the high-frequency mid-wavenumber region, visible as "diagonal stripes". These were attenuated in the complex-valued network in Figure 10b, but are partially present in the real-valued reconstruction in Figure 10c.

7. Discussion

We evaluated the outputs of the real-valued and complex-valued neural networks. All auto-encoder outputs are blurred to different degrees and denoised. The denoising effect of the seismic was most visible in

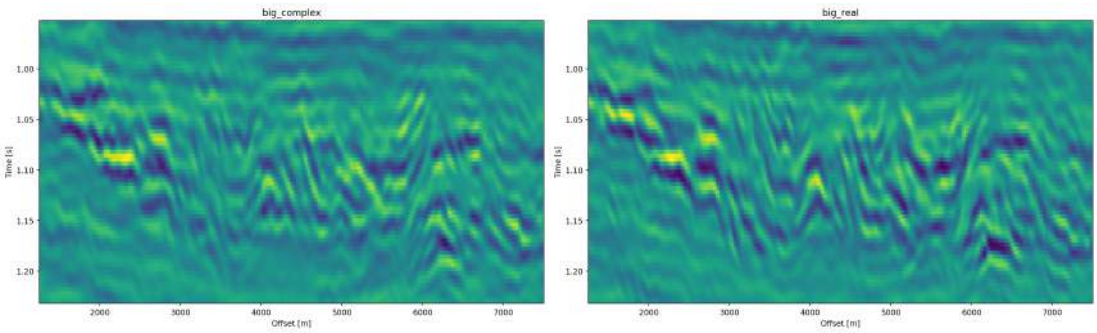


(a) Ground Truth



(b) Small Complex Network top Patch

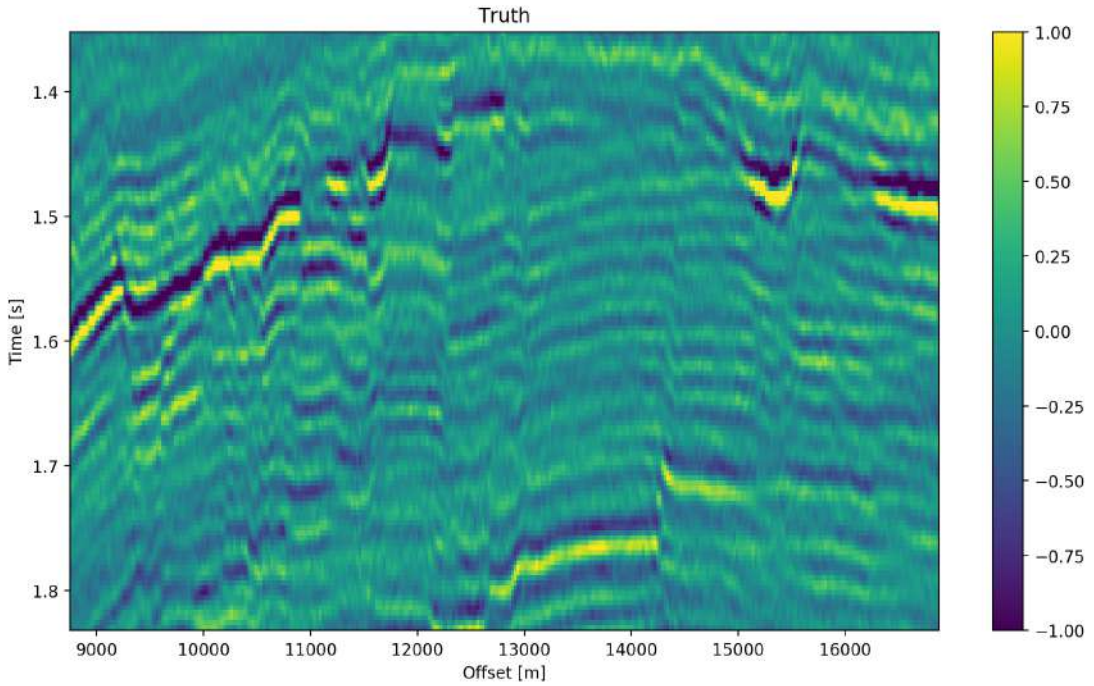
(c) Small Real Network top Patch



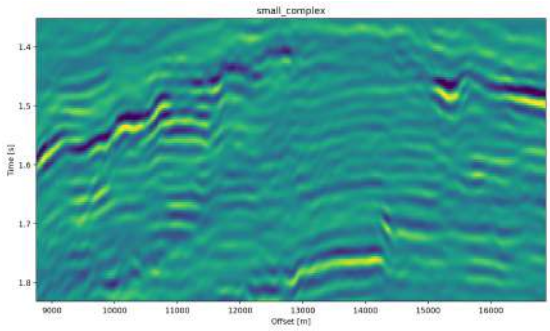
(d) Large Complex Network top Patch

(e) Large Real Network top Patch

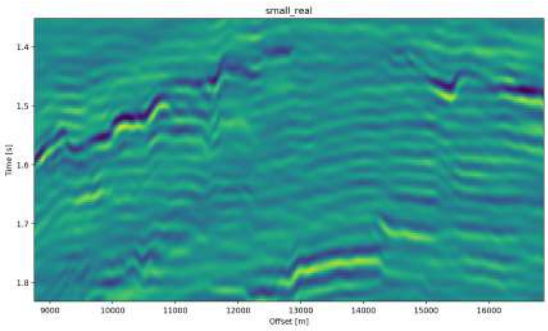
Figure 8: Evaluation on top Noise Patch



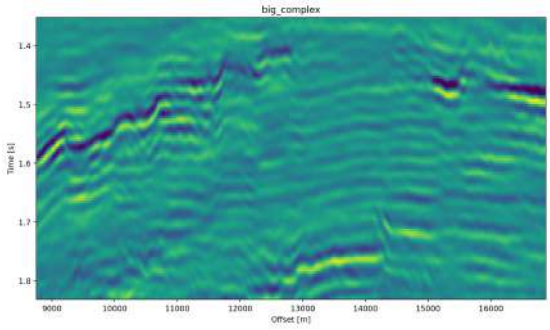
(a) Ground Truth



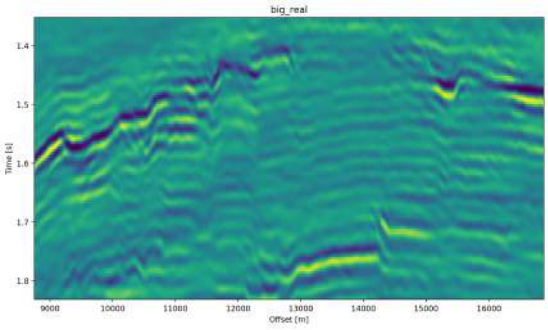
(b) Small Complex Network bottom Patch



(c) Small Real Network bottom Patch

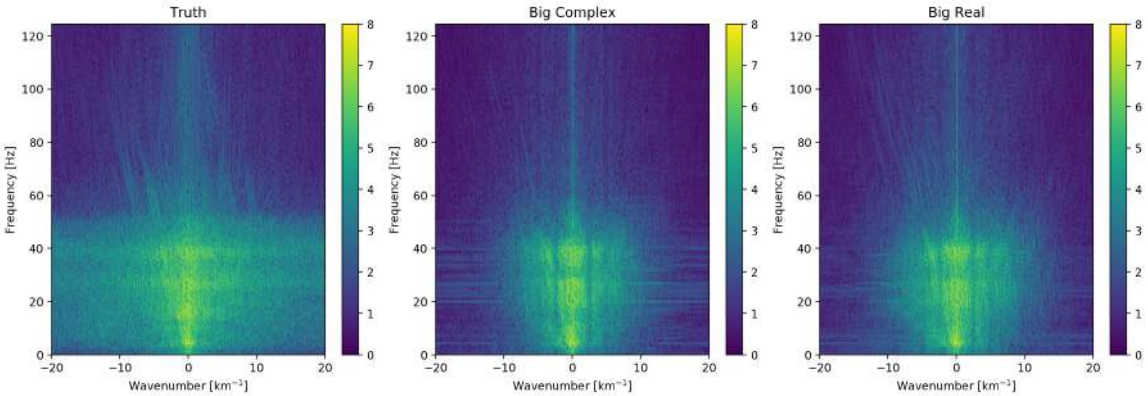


(d) Large Complex Network bottom Patch



(e) Large Real Network bottom Patch

Figure 9: Evaluation on bottom Noise Patch



(a) Ground Truth (b) Large Complex Network (c) Large Real Network

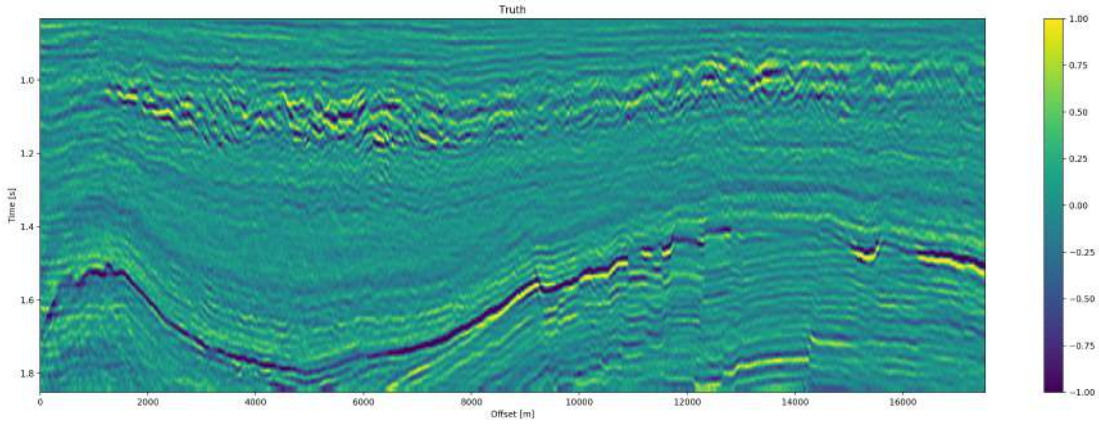
Figure 10: FK domain of full seismic data.

the frequency band below 50 Hz. Additionally, some scattered high-frequency energy was attenuated by the networks.

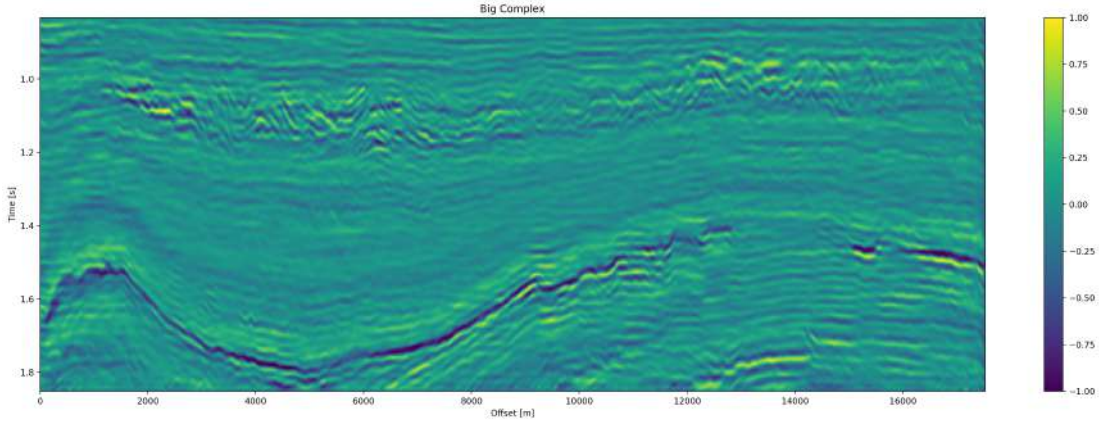
The largest differences of the outputs in real-valued and complex-valued networks can be observed in discontinuous areas. Particularly, the faulted blocks in the top quarter and in the bottom center of the seismic section show inconsistencies. The real-valued network smooths over discontinuities and steep reflectors. Fault lines are imaged better in the complex-valued network output.

In seismic data processing, including phase information stabilizes discontinuities and disambiguates cycle-skipping in horizons. This could be observed in the network performance and reconstruction. The increase in performance of the real-valued networks was significant (7.0 % RMS), while the complex-valued networks already had an acceptable performance on the smaller network architecture (2.6 % RMS). We provide the complex-valued networks with a bias towards learning phase information, by providing the Hilbert transformed analytical trace, while the real-valued network needs to learn this information implicitly from the data itself. Considering, that during the training, the complex network evaluates both the real-valued seismic, which we primarily care about in addition to the complex-valued component, we can see how the losses in Figure 5 differ from the real-valued networks.

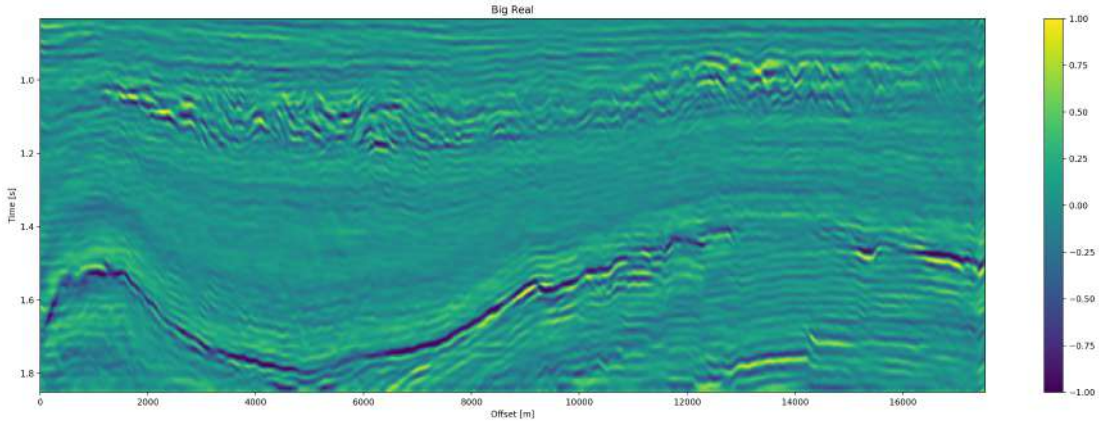
The largest network with 790,945 trainable parameters quantitatively performed the best on the reconstruction of the data. However, analysis of the reconstructed seismic shows, that while the high-amplitude regions are reconstructed to higher fidelity, discontinuous sections may be smeared by the real-valued network. The real-valued network that was matched to contain as many filters for the real-valued component of the seismic as the large complex-valued network, did not perform well. Furthermore, the smaller complex-valued network with 100,226 parameters that contains as many filter maps as the real-valued network in total, and half the trainable parameters, outperformed the smaller real-valued network across all test cases.



(a) Ground Truth



(b) Large Complex Network



(c) Large Real Network

Figure 11: Evaluation on full seismic data.

8. Conclusion

The inclusion of phase-information leads to a better representation of seismic data in convolutional neural networks. Complex-valued networks perform consistently, where real-valued networks have to learn phase-representations through implicit correlation, which requires larger networks. We show that complex trace information in deep neural networks improves the imaging of discontinuities as well as steep reflectors, particularly in chaotic seismic textures that are smoothed by real-valued neural networks of the same size.

We show that convolutional neural networks can perform lossy compression on seismic data, where the reconstruction error is dependent on both network architecture and implementation details, like providing explicit phase information. During this compression, noise and scattered energy get attenuated. The real-valued network is prone to introduce steeply dipping artifacts in the reconstruction.

The stabilization of the reconstruction can be useful in seismic applications. While automatic seismic interpretation may benefit from the inclusion of information on discontinuities, we see the main application to be lossy seismic compression. The open source tool developed to make this research possible, enables further research and development of complex-valued solutions to non-stationary physics problems that benefit from explicit phase information.

The research shows that a change as small as 2.5 % in RMS can change the reconstruction from being acceptable to very smeared to a geoscientist. This touches on the fact that better metrics to evaluate computer vision tasks in geoscience are necessary. Additionally, these tasks have to be noise-robust and while amplitude-preserving be outlier robust too. Moreover, more research in the frequency dimming of bands in the network reconstruction is necessary.

Overall, the computational memory footprint of the complex convolution is higher than real-valued convolutional neural networks comparing singular convolutional operations. A significant increase in depth and width of networks to obtain an acceptable result in real-valued neural network to implicitly learn the phase information is necessary. The complex-valued networks an 8^{th} of the size already performs well, suggesting that expert domains that contain beneficial information in the phase of signals, could benefit from applying complex convolutional networks.

9. Acknowledgments

We thank Andrew Ferlitsch for his valuable insights. The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. We thank DTU Compute for access to the GPU Cluster.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser,

L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: http://tensorflow.org/. software available from tensorflow.org.

Alaudah, Y., Michalowicz, P., Alfarraj, M., AlRegib, G., 2019. A machine learning benchmark for facies classification. arXiv preprint arXiv:1901.07659 .

Barnes, A.E., 2007. A tutorial on complex seismic trace analysis. Geophysics 72, W33–W43.

Chollet, F., et al., 2015. Keras. https://keras.io.

Dramsch, J.S., Lüthje, M., 2018a. Deep-learning seismic facies on state-of-the-art cnn architectures, in: SEG Technical Program Expanded Abstracts 2018. Society of Exploration Geophysicists, pp. 2036–2040.

Dramsch, J.S., Lüthje, M., 2018b. Information theory considerations in patch-based training of deep neural networks on seismic time-series, in: First EAGE/PESGB Workshop Machine Learning, EAGE Publications BV. URL: https://doi.org/10.3997/2214-4609.201803020, doi:10.3997/2214-4609.201803020.

Griffin, D., Lim, J., 1984. Institute of Electrical and Electronics Engineers. pp. 236–243. URL: https://doi.org/10.1109/icassp.1983.1172092, doi:10.1109/icassp.1983.1172092.

Hinton, G.E., Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. science 313, 504–507.

Hirose, A., Yoshida, S., 2012. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. IEEE Transactions on Neural Networks and Learning Systems 23, 541–551. URL: https://doi.org/10.1109/tnnls.2012.2183613, doi:10.1109/tnnls.2012.2183613.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 .

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, pp. 1097–1105.

LeCun, Y., Haffner, P., Bottou, L., Bengio, Y., 1999. Object recognition with gradient-based learning, in: Shape, contour and grouping in computer vision. Springer, pp. 319–345.

Liner, C.L., 2002. Phase, phase, phase. The Leading Edge 21, 456–457. URL: https://doi.org/10.1190/1.1885500, doi:10.1190/1.1885500.

Mavko, G., Mukerji, T., Dvorkin, J., 2003. The rock physics handbook.

Mehri, S., Kumar, K., Gulrajani, I., Kumar, R., Jain, S., Sotelo, J., Courville, A.C., Bengio, Y., 2016. Samplernn: An unconditional end-to-end neural audio generation model. CoRR abs/1612.07837. URL: http://arxiv.org/abs/1612.07837, arXiv:1612.07837.

Miyauchi, M., Seki, M., Watanabe, A., Miyauchi, A., 1993. Interpretation of optical flow through complex neural network, in: New Trends in Neural Computation. Springer Berlin Heidelberg, pp. 645–650. URL: https://doi.org/10.1007/3-540-56798-4_215, doi:10.1007/3-540-56798-4_215.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A.W., Kavukcuoglu, K., 2016. Wavenet: A generative model for raw audio. CoRR abs/1609.03499. URL: http://arxiv.org/abs/1609.03499, arXiv:1609.03499.

van den Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., van den Driessche, G., Lockhart, E., Cobo, L.C., Stimberg, F., Casagrande, N., Grewe, D., Noury, S., Dieleman, S., Elsen, E., Kalchbrenner, N., Zen, H., Graves, A., King, H., Walters, T., Belov, D., Hassabis, D., 2017. Parallel wavenet: Fast high-fidelity speech synthesis. CoRR abs/1711.10433. URL: http://arxiv.org/abs/1711.10433, arXiv:1711.10433.

Paganini, M., de Oliveira, L., Nachman, B., 2017. CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks arXiv:1705.02355.

Prenger, R., Valle, R., Catanzaro, B., 2018. Waveglow: A flow-based generative network for speech synthesis. CoRR abs/1811.00002. URL: http://arxiv.org/abs/1811.00002, arXiv:1811.00002.

Purves, S., 2014. Phase and the hilbert transform. The Leading Edge 33, 1164–1166. URL: https://doi.org/10.1190/tle33101164.1, doi:10.1190/tle33101164.1.

Roden, R., Sepúlveda, H., 1999. The significance of phase to the interpreter: Practical guidelines for phase analysis. The Leading Edge 18, 774–777. URL: https://doi.org/10.1190/1.1438375, doi:10.1190/1.1438375.

Sarroff, A.M., 2018. Complex Neural Networks for Audio. Technical Report TR2018-859. Dartmouth College, Computer Science. Hanover, NH. URL: http://www.cs.dartmouth.edu/~trdata/reports/TR2018-859.pdf.

Sarroff, A.M., Shepardson, V., Casey, M.A., 2015. Learning representations using complex-valued nets. CoRR abs/1511.06351. URL: http://arxiv.org/abs/1511.06351, arXiv:1511.06351.

Scarpiniti, M., Vigliano, D., Parisi, R., Uncini, A., 2008. Generalized splitting functions for blind separation of complex signals. Neurocomputing 71, 2245–2270. URL: https://doi.org/10.1016/j.neucom.2007.07.037, doi:10.1016/j.neucom.2007.07.037.

Suksmono, A.B., Hirose, A., 2002. Adaptive noise reduction of InSAR images based on a complex-valued MRF model and its application t o phase unwrapping problem. IEEE Transactions on Geoscience and Remote Sensing 40, 699–709. URL: https://doi.org/10.1109/tgrs.2002.1000329, doi:10.1109/tgrs.2002.1000329.

Taner, M.T., Koehler, F., Sheriff, R., 1979. Complex seismic trace analysis. Geophysics 44, 1041–1063.

Trabelsi, C., Bilaniuk, O., Zhang, Y., Serdyuk, D., Subramanian, S., Santos, J.F., Mehri, S., Rostamzadeh, N., Bengio, Y., Pal, C.J., 2017. Deep complex networks. arXiv preprint arXiv:1705.09792 .

Yilmaz, Ö., 2001. Seismic data analysis. volume 1. Society of exploration geophysicists Tulsa, OK.

5.3 Information Theory Considerations in Patch-based Training of Deep Neural Networks on Seismic Time-Series

Abstract: Recent advances in machine learning relies on convolutional deep neural networks. These are often trained on cropped image patches. Pertaining to non-stationary seismic signals this may introduce low frequency noise and non-generalizability.

Key points:

- Convolutional Neural Networks use windowed convolutions
- Non-stationary data can ”seem” offset from zero
- Results in erroneous bias within network

J. S. Dramsch and M. Lüthje (2018d). “Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series”. In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 5. EAGE. DOI: 10.3997/2214-4609.201803020. URL: https://doi.org/10.3997/2214-4609.201803020

Introduction

Sampling in physics-based applications and digital signal processing has long been recognised as an essential constraint. The Nyquist-Shannon theorem is the most prominent information theorem that prevents aliasing in seismic data (Seibt, 2006). Sampling has to be considered an essential part of a machine learning pipeline to avoid the implicit bias of learnt decision boundaries and joint distributions.

Machine Learning algorithms, particularly deep convolutional neural networks (CNN) often learn on patches of data. In many applications, the dynamic range of the data is additionally converted from 32-bit floats to 8-bit integers. This loss of dynamic range often speeds up training of networks and stabilises convergence at the loss of accuracy. However, investigations into precision have shown that this effect may be negligible (Holi and Hwang, 1993). Patch-based image training in machine learning usually takes smaller windows of data. The ImageNet challenge (Deng et al., 2009) provides 256x256 pixel images, which sets standards for many machine learning architectures.

Theory

Seismic traces are often sampled at 4ms and contain several hundred to thousands of samples. The Nyquist-Shannon theorem applies to high-frequency bounds only. However, we propose that a lower bound has to be adhered to when applying real-valued transformations to data before reconstruction. Low-frequency aliasing can be seen as a DC offset, where DC is the value at 0 Hz. This effect has been studied in non-stationary signals in applications such as seismic frequency decomposition (Chakraborty and Okaya 1995).

In statistical learning, many applications learn implicit joint distributions of the data. These are often approximated by multivariate distributions or transformations that operate solely on real-valued signals instead of complex signals (Hirose, 2003). This is equivalent to a mean shift of the data, as well as noise of the mean and may hinder convergence of the algorithms and diminish results. Inference on images that can appropriately sample low frequencies, due to a larger size, could lead to non-generalizability of the data due to implicit bias, which is the antithesis of machine learning.

We propose a low-frequency boundary, which follows the Nyquist-Shannon sampling theorem. With $f_{ny} = \frac{1}{2T}$, where T is the maximum period resolvable in the time series. This is due to the fact that we treat cutouts of a non-stationary signal as representative of the entire series and therefore, have to infer stationarity within the available bandwidth.

Example: Neural Network - Single Neuron

Neurons in neural networks are described by the activation $\sigma(w \cdot x + b)$, where w is the network weights, x is the input data, b is the network bias, and sigma is a non-linear activation function. A common non-linear activation is the rectified linear unit (RELU) $\sigma(x) = max(0, x)$. Considering the inference stage, the network weights w and biases b are fixed, x is the only variable parameter. Learning on a mean-shift of q of an arbitrary distribution over x leads to $\sigma(w \cdot (x + q) + b)$, which increases the neuron response by q, weighted by w. At inference, the mean-shift over larger inference data disappears, introducing an additional bias of $w \cdot q$ before non-linear activation. This training bias may lead to prediction errors of the neuron and consequently the full neural network.

Example: Dutch F3 Seismic data

We use a randomly selected trace from the Dutch F3 dataset. The total recording time is 4 seconds with 1001 samples sampled at 4 ms. The sampling interval of 4ms allows for a maximum frequency of 125 Hz. We compare the reconstruction of the signal from the real part of the frequency spectrum for non-overlapping patches. The frequency content of real-valued stationary traces would be similar, whether a trace is split into parts or whole.

The frequency content in Figure 1 shows that properly tapered data introduces a DC offset and the phase spectrum cannot be reconstructed fully. For a window of 101 samples at 4 ms, we get the lower Nyquist frequency of ~12.5 Hz. A patch of 256 samples at 4ms has the lower bound of ~5 Hz. We propose that a high-pass filter at training may improve convergence. Transfer learning on larger patches with fewer epochs then recovers low-frequency information, while keeping training times attainable.

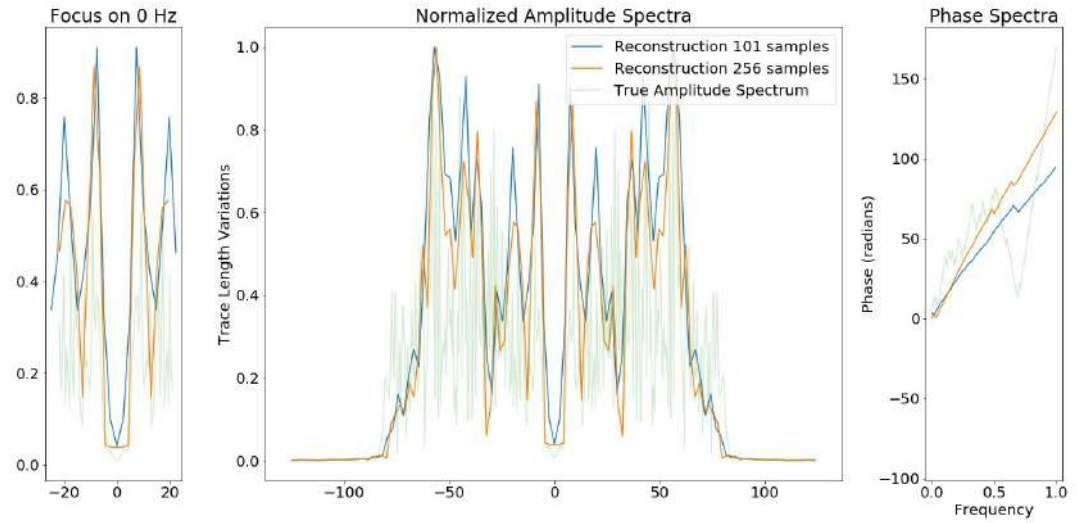


Figure 1 *We present different sizes of cutouts, with 101 and 256 samples respectively. In the middle, the full normalised amplitude spectra are presented. On the right, the according phase spectra are presented. On the left, we focus on the frequency content of the amplitude spectra around 0 Hz. The cutouts were Hanning tapered, however, a clear DC offset appears with decreasing patch size.*

Conclusions

We investigate the frequency content in non-overlapping patch-based seismic data. Non-overlapping patches may introduce low-frequency noise that translates to a mean-shift of learnt distributions. Further investigations into frequency responses of Convolutional Neural Networks (CNN) and the computation thereof, which is common in the frequency domain, should be undertaken. The authors note that signal processing paradigms apply to image-based CNNs and tapering of time-series before Fourier transformation is essential.

Acknowledgements

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. The authors thank Matthias Schneider for fruitful discussion and dGB for providing the F3 dataset.

References

Avijit Chakraborty and David Okaya (1995). "Frequency-time decomposition of seismic data using wavelet-based methods." GEOPHYSICS, 60(6), 1906-1916

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (pp. 248-255). Ieee.

Hirose, A. (2003). Complex-Valued Neural Networks: An Introduction. In *Complex-Valued Neural Networks: Theories and Applications* (pp. 1-6).

Holi, J. L., & Hwang, J. N. (1993). Finite precision error analysis of neural network hardware implementations. *IEEE Transactions on Computers*, (3), 281-290.

Seibt, P. (2006). *Algorithmic Information Theory*. Springer.

5.4 Deep learning seismic facies on state of the art CNN architectures

Abstract: We explore propagation of seismic interpretation by deep learning in stacked 2D sections. We show the application of state-of-the-art image classification algorithms on seismic data. These algorithms were trained on big labeled photograph databases. We use transfer learning to benefit from pre-trained networks and evaluate their performance on seismic data.

Key points:

- Pre-trained Convolutional Neural Networks transfer to seismic
- Outperforms smaller CNNs trained from scratch
- Enables training in data sparse environments

J. S. Dramsch and M. Lüthje (2018c). "Deep-learning seismic facies on state-of-the-art CNN architectures". In: *SEG Technical Program Expanded Abstracts 2018*. Published, Chapter 5. Society of Exploration Geophysicists, pp. 2036–2040. DOI: 10.1190/segam2018-2996783.1. URL: https://doi.org/10.1190/segam2018-2996783.1

Deep learning seismic facies on state-of-the-art CNN architectures

Jesper S. Dramsch, Technical University of Denmark, and Mikael Lüthje, Technical University of Denmark*

SUMMARY

We explore propagation of seismic interpretation by deep learning in stacked 2D sections. We show the application of state-of-the-art image classification algorithms on seismic data. These algorithms were trained on big labeled photograph databases. We use transfer learning to benefit from pre-trained networks and evaluate their performance on seismic data.

INTRODUCTION

Seismic interpretation is often dependent on the interpreters experience and knowledge. While deep learning cannot replace expert knowledge, we explore the accuracy of convolutional networks in interpreting seismic data to support human interpretation.

In the 1950s neural networks started as a simple direct connection of several nodes in an input layer to several nodes in an output layer (Widrow and Lehr, 1990). In geophysics this puts us to the introduction of seismic trace stacking (Yilmaz, 2001). In 1989 the first idea of a convolutional neural network was born (Lecun, 1989) and back-propagation was formalized as an error-propagation mechanism (Rumelhart et al., 1988). In 2012 the paper (Krizhevsky et al., 2012) propelled the field of deep learning forward implementing essential components, namely GPU training, ReLu activation functions (Dahl et al., 2013) and dropout (Srivastava et al., 2014). They outperformed previous models in the ImageNet challenge (Deng et al., 2009) by almost halving the prediction error. Waldeland and Solberg (2016) showed that neural networks can be used to classify salt diapirs in 3D seismic data. Charles Rutherford Ildstad (2017) generalized this work to nD and beyond two classes of salt and "else".

The task of automatic seismic interpretation can be equated to dense object detection (Lin et al., 2017) or semantic segmentation. These tasks are currently best solved by Mask R-CNN architectures (Long et al., 2015). Statoil has used U-Nets for automatic seismic interpretation. Yet, classification networks can be used for semantic segmentation, but are significantly slower. The benefit is a testable example of generalization of pre-trained networks form photographic data to seismic images. As well as, a testable framework for choosing hyperparameters for neural networks on seismic data.

Deep learning relies heavily on vast amounts of labeled data to train on initially. However, the features learned from these networks can often be transferred to adjacent problem spaces (Baxter, 1998). Often these transfer learning tasks are tested on photographs rather than seismic or medical imaging tasks. The aim of this study is to evaluate state-of-the-art pre-trained networks in the task of automatic seismic interpretation. We compare three convolutional neural networks of increasing complexity in the task of supervised automatic seismic interpretation. We evaluate these tasks qualitatively and quantitatively.

METHODS

The neural networks in this study learn supervised. The features were published alongside the open source framework MalenoV and describe nine seismic facies in the open F3 data set. The classes describe steep dipping reflectors, salt intrusions, low coherency regions, low amplitude dipping reflectors, high amplitude regions continuous high amplitude regions and grizzly amplitude patterns presented in figure 3. Additionally, a catch-all "else" region are picked. In this approach we chose Keras (Chollet et al., 2015) with a Tensorflow (Abadi et al., 2015) backend on a K5200 GPU at DHRTC. Keras is a powerful high level abstraction of tensor arithmetics. Tensorflow is an open source numerical computation library on static graphs. We train 2D convolutional neural networks (CNN) of varying depth on seismic slices to propagate single slice interpretations to a volume. CNNs are highly flexible models for computer vision tasks.

Network one depicted in figure 2 was developed by Waldeland and Solberg (2016) to identify salt bodies in 3D seismic data. Three layers are fully connected for classification. The network uses a kernel of 5 by 5 pixels for convolution and a stride of 2 for down-sampling. We use the Adam optimizer and cross-categorical entropy as a loss function. The Adam optimizer is an extension to stochastic gradient descent (SGD) that implements adaptive learning rates and bias correction (Ruder, 2016). We add dropout and batch normalization to the network. These methods improve regularization and prevent overfitting. Furthermore, we use early-stopping to prevent overfitting the model by over-training. We chose two metrics to monitor in the training and validation sets, namely mean absolute error and accuracy. The Waldeland CNN is relatively shallow compared to modern deep learning networks with 95,735 parameters to optimize for.

Network two is the VGG16 network (Simonyan and Zisserman, 2014) by the Visual Geometry Group. It contains 16 layers and 1,524,2605 parameters. 13 of these layers ore convolutional layers with a 3x3 kernel. Convolutional blocks are interspersed with max-pooling layers for down-sampling. The last three layers are fully connected layers for classification. The VGG16 architecture was proposed for the ImageNet challenge in 2013. It is widely used for it's simplicity in teaching and it's generalizability in transfer learning tasks.

Network three is the ResNet50 architecture by Microsoft. The network consists of 50 layers with 2,361,6569 parameters. It implements a recent development, called residual blocks. These residual blocks add a skip- or identity-connection around a stack of 1x1, 3x3, 1x1 convolutional layers (He et al., 2016). The 1x1 are identity convolutions, used for down- and subse-

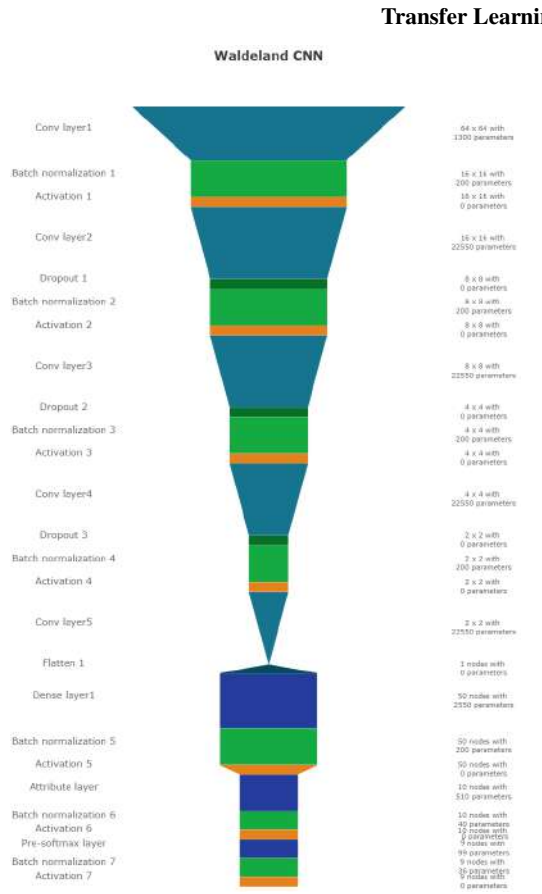


Figure 1: Waldeland CNN architecture. Input at the Top. Softmax Classification Layer on bottom. Width of objects shows `log` of spatial extent of layer. Height shows `log` of complexity of layer. The layers are color coded to show similar purpose.

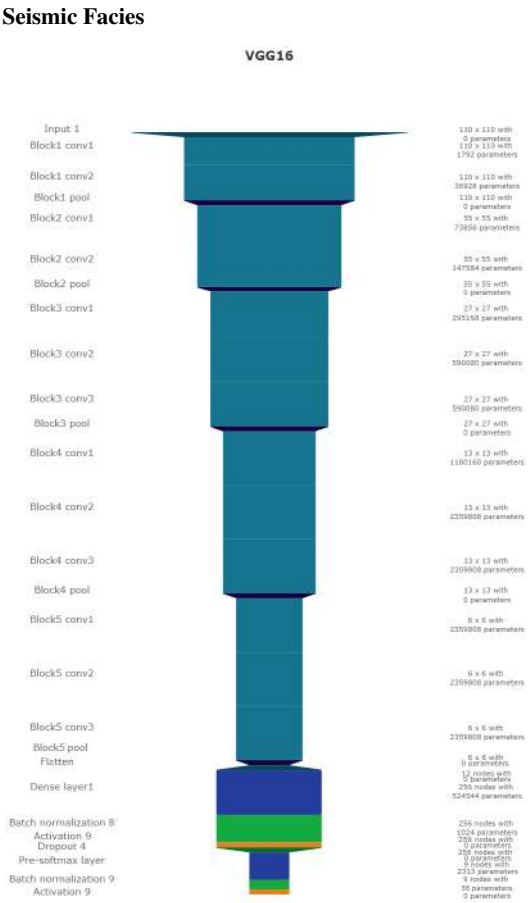


Figure 2: VGG16 architecture. Same visualization as figure 2

quent up-sampling to decrease the computational cost of very deep CNNs. The convolutional layers are followed by one fully connected layer for classification.

All networks use rectified linear units (ReLu) as neural activation. The last layer uses Softmax as activation to output a probability for each class. Training both VGG16 and the ResNet50 end to end would be very expensive. These models have been trained on big labeled data that are not available in geoscience. However, transfer learning enables us to use pre-trained networks on very different tasks. In transfer learning, we use the learned weights of the networks and replace the fully connected layers. These untrained layers are specific to our task and have to be fine-tuned to the data. This process is very fast and requires little data. We fine-tune an entire network on one sparsely interpreted 2D seismic slice. For the fine-tuning process, we replace the Adam optimizer by a classic SGD optimizer with lower learning rate, very low weight decay and Nesterov momentum. We still use early-stopping on validation loss and cross-categorical entropy.

We added the same fully connected layer architecture to VGG16 and ResNet50 that Waldeland added to their architecture. Therefore, we test if pre-trained convolution kernels are fit to recognize texture features in seismic data. We set up a validation set to quantify the accuracy of our networks on previously unseen data. Additionally, we set up a prediction pipeline to populate each one 2D inline and crossline of the seismic data to qualitatively visualize the prediction capability of the networks. The labels for the supervised interpretation are taken from the MalenoV interpretation by ConocoPhillips, shown in figure 3.

Network	Run	Loss	MAE	Acc
Waldeland CNN	Training	0.001	0.000	100.0%
	Test	0.003	0.000	99.9%
VGG16	Training	0.010	0.005	99.8%
	Test	0.127	0.026	100.0%
ResNet50	Training	0.011	0.001	100.0%
	Test	14.166	0.195	12.1%

Table 1: Training and Test scores on Networks. Test scores are prediction results on a labeled hold-out data set. Mismatch of test and training scores indicates over-fitting.

Transfer Learning Seismic Facies

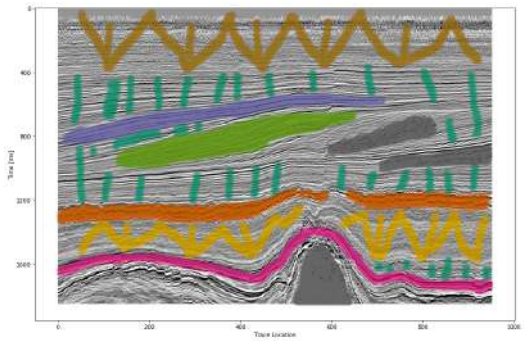


Figure 3: Labeled data set on one 2D inline slice. Color interpretation: Low coherency (brown), Steep dipping reflectors (gray), low amplitude dipping reflectors (grass green), continuous high amplitude regions (blue), grizzly (orange), low amplitude (yellow), high amplitude (magenta), salt intrusions (gray), else (turquoise).

RESULTS

We use the open Dutch F3 data set to calibrate our predictions. Crossline 339 has been interpreted by ConocoPhillips and made available freely. We show results of crossline slice 500. We have used the same plotting parameters for both either results, both have been generated programatically, without human intervention. Figure 4(a) shows the prediction of the Waldeland CNN at every location of the 2D slice based on a 65 x 65 patch of the data. Border patches were zero padded. We see clear patches for the low coherency region in brown. The low amplitude dipping (grass green) region has been reproduced well, however some regions at $t \approx 1080$ *ms* have been marked incorrectly, where two seismic packages meet. This faulty region also contains patches that were interpreted as low amplitude region (yellow). While this may be a low amplitude region, we expect the packages to be largely continuous, which leaves this interpretation as questionable at best. The gray area was reproduced well, however it was marked as salt body in the original manuscript, this would be incorrect here. We see the grizzly amplitude pattern (orange) and the low amplitude (yellow) regions are well-defined and separated. The underlying package of high amplitudes has been identified will. However, between location 600 - 800 the top part was marked as "else" (turquoise), which undesirable but correct, judging from the texture. Here, retraining would be possible by feeding this relabeled region to the network. Below this region, the networks predictions become erratic. The classification is blocky between grizzly and salt with "else" interspersed. However, the edges will often give problems due to the padding. Around location 800 high amplitudes (orange) have been mislabeled as grizzly amplitudes.

The VGG16 network classification is shown in figure 4(b). The network performs similar to the Waldeland CNN in figure 4(a), however some key differences will be pointed out. The separation of low coherency and the "else" region around $t \approx 400$ *ms* is less defined and, therefore, worse. The coherency of low amplitude dipping (grass green) and high amplitude continuous (blue) is worse in the region around location 280, $t \approx 800$ *ms*. This might be due to higher sensitivity to declines in seismic quality. Below $t \approx 1000$ *ms* the "else" region is free from differing patches, in contrast, the Waldeland CNN interspersed two other classes in this region. VGG16 also classifies some "else" regions in the high amplitude (magenta) region between location 600-800. The area around location 200 below the high amplitude (magenta) region is also blocky, although less so. The misclassification of the bottom high amplitude (magenta) region as grizzly (orange) is less pronounced in the VGG16 interpretation. It is present toward the bottom left corner.

The results of the ResNet50 are not shown. The network classifies all seismic facies as "else". This indicates that the network is overfitting the data. This is supported by the numeric results presented in table 1. The network training error indicates a perfect fit to the data, whereas the test score is unseen data with labels to evaluate the performance of networks on unseen data. While both the Waldeland CNN and VGG16 perform well, the ResNet50 performs very poorly.

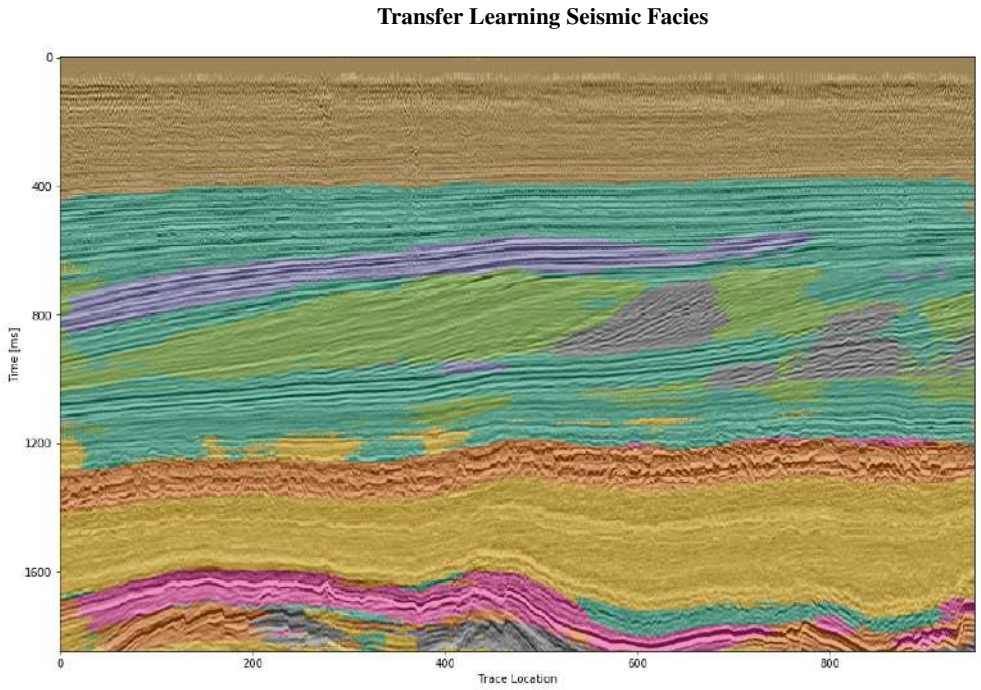
CONCLUSION

Convolutional neural networks show good results for propagating interpretations through seismic cubes. The pre-trained VGG16 CNN has shown very good results in adapting to seismic texture identification. Transfer learning was fast and the results are similar to the shallower Waldeland CNN. Both networks have trade-offs in the misclassification and can be improved upon.

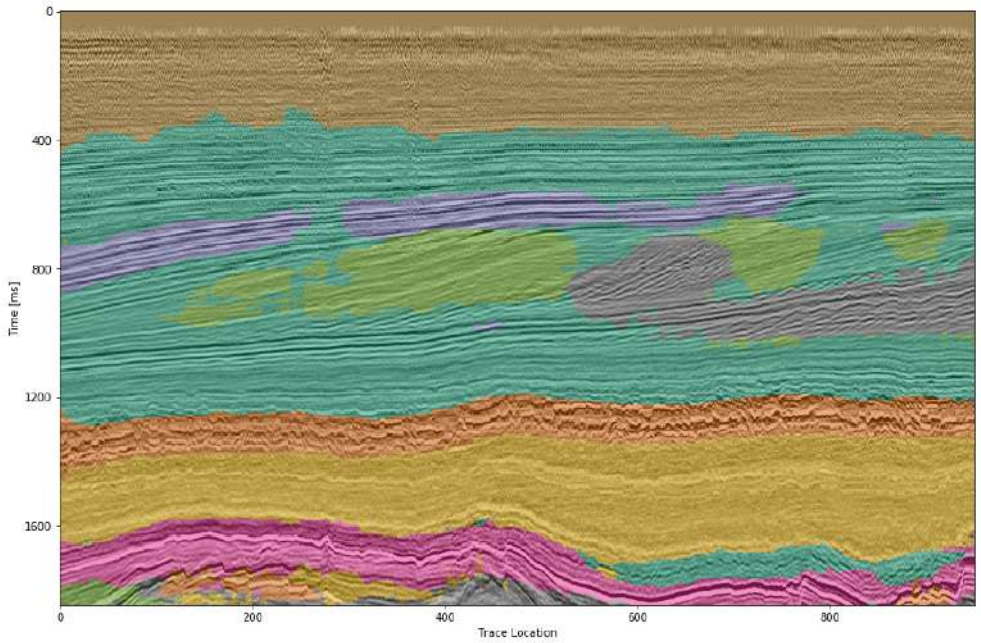
The ResNet50 was shown to be ineffective on transfer learning seismic data with pre-trained weights. This is in accordance with results from other attempts at transfer learning. The ResNet filters are more specific to photography and transfer poorly to other data sources, where the VGG learned features prove to be more general to computer vision tasks. More complicated architectures may perform well, trained directly with the according data, but they learn specific features fit for the problem space that do not transfer well.

ACKNOWLEDGMENTS

The authors would like to thank the DHRTC and DUC for their continued support. We thank Colin MacBeth, Peter Bormann, Sebastian Tølbøll Glavind, Lukas Mosser and the "Software Underground" community for great discussion and support with MalenoV and ConocoPhillips for making the data and software freely available. We also thank Agile Scientific for great tutorials at the intersection of Python and geoscience. We thank dgb for providing the F3 data set.



((a)) Waldeland CNN automatic interpretation of crossline 500.



((b)) VGG16 automatic interpretation of crossline 500.

Figure 4: Automatic seismic interpretation with CNNs. Color interpretation: Low coherency (brown), Steep dipping reflectors (gray), low amplitude dipping reflectors (grass green), continuous high amplitude regions (blue), grizzly (orange), low amplitude (yellow), high amplitude (magenta), salt intrusions (gray), else (turquoise).

REFERENCES

Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, 2015, TensorFlow: Large-scale machine learning on heterogeneous systems. (Software available from tensorflow.org).

Baxter, J., 1998, Theoretical models of learning to learn, *in* Learning to learn: Springer, 71–94.

Charles Rutherford Ildstad, P. B., 2017, MalenoV. Machine learning of Voxels.

Chollet, F., et al., 2015, Keras: https://github.com/fchollet/keras.

Dahl, G. E., T. N. Sainath, and G. E. Hinton, 2013, Improving deep neural networks for LVCSR using rectified linear units and dropout: Presented at the 2013 IEEE International Conference on Acoustics Speech and Signal Processing, IEEE.

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, 2009, ImageNet: A Large-Scale Hierarchical Image Database: Presented at the CVPR09.

He, K., X. Zhang, S. Ren, and J. Sun, 2016, Deep residual learning for image recognition: Proceedings of the IEEE conference on computer vision and pattern recognition, 770–778.

Krizhevsky, A., I. Sutskever, and G. E. Hinton, 2012, ImageNet Classification with Deep Convolutional Neural Networks, *in* Advances in Neural Information Processing Systems 25: Curran Associates, Inc., 1097–1105.

Lecun, Y., 1989, Generalization and network design strategies, *in* Connectionism in perspective: Elsevier. (Accessed on Mon, November 20, 2017).

Lin, T.-Y., P. Goyal, R. Girshick, K. He, and P. Dollár, 2017, Focal loss for dense object detection: arXiv preprint arXiv:1708.02002.

Long, J., E. Shelhamer, and T. Darrell, 2015, Fully convolutional networks for semantic segmentation: Proceedings of the IEEE conference on computer vision and pattern recognition, 3431–3440.

Ruder, S., 2016, An overview of gradient descent optimization algorithms: arXiv preprint arXiv:1609.04747.

Rumelhart, D., G. Hinton, and R. Williams, 1988, Learning Internal Representations by Error Propagation, *in* Readings in Cognitive Science: Elsevier, 399–421.

Simonyan, K., and A. Zisserman, 2014, Very deep convolutional networks for large-scale image recognition: arXiv preprint arXiv:1409.1556.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014, Dropout: A Simple Way to Prevent Neural Networks from Overfitting: Journal of Machine Learning Research, **15**, 1929–1958.

Waldeland, A., and A. Solberg, 2016, 3D Attributes and Classification of Salt Bodies on Unlabelled Datasets: Presented at the 78th EAGE Conference and Exhibition 2016, EAGE Publications BV.

Widrow, B., and M. Lehr, 1990, 30 years of adaptive neural networks: perceptron Madaline, and backpropagation: Proceedings of the IEEE, **78**, 1415–1442.

Yilmaz, O., 2001, Seismic Data Analysis: Society of Exploration Geophysicists.

CHAPTER 6

Deep Neural Networks for 4D Seismic Inversion

This chapter is comprised of two accepted conference papers that describe a Deep Neural Network for pressure-saturation inversion of 4D seismic data.

Papers:

J. S. Dramsch, G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019d). "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data". In: *Second EAGE Workshop Practical Reservoir Monitoring 2019*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201900028

J. S. Dramsch, G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019e). "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201901967. URL: https://doi.org/10.3997/2214-4609.201901967

6.1 Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion

Abstract: Geoscience data often have to rely on strong priors in the face of uncertainty. Additionally, we often try to detect or model anomalous sparse data that can appear as an outlier in machine learning models. These are classic examples of imbalanced learning. Approaching these problems can benefit from including prior information from physics models or transforming data to a beneficial domain. We show an example of including physical information in the architecture of a neural network as prior information. We go on to present noise injection at training time to successfully transfer the network from synthetic data to field data.

Key points:

- Deep Neural Network to invert seismic for pressure-saturation data
- Compared to Bayesian inversion
- Indicators for good performance:

  Context-unaware network results have spatial consistency

  Values unconstrained but predict correct range

  Areas of effect match prediction
- Training on simulation data transfers to field data

J. S. Dramsch, G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019e). "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201901967. URL: https://doi.org/10.3997/2214-4609.201901967

2019 EAGE ANNUAL
81ST CONFERENCE + EXHIBITION
LONDON | UNITED KINGDOM

Introduction

Physics in machine learning often relies on transformations of data to beneficial domains and simulating additional data. Karpatne et al. (2017) show a physics-guided approach to model lake temperatures with neural networks. Schütt et al. (2017) use deep neural networks to model molecule energies and de Oliveira et al. (2017) employ a special architecture to capture scatter patterns in high-energy physics. When building deep learning pipelines, we can make informed choices in data modeling, but also build neural networks to maximize information gain on the available data. Ulyanov et al. (2018) has shown that the network architecture itself can be used as prior in machine learning. These approaches translate well to geoscience, where strong priors are often necessary to inform decisions.

Deep learning has revolutionized machine learning by replacing the feature generation and augmentation step by learned internal representations of features that maximize information gain. On image data analysis of these neural network filters have shown close relations to edge filters and color separators (Grün et al., 2016). Dramsch and Lüthje (2018) have shown that these filters translate well to seismic data. However, classic feed-forward neural networks do not have the benefit of learning filters. However, these neural networks benefit from recent improvements for regularization (Ioffe and Szegedy, 2015), non-saturating and non-vanishing gradients (He et al., 2015), and training on GPUs.

Neural networks for inversion of seismic data have a long history (Roeth and Tarantola, 1994). In Dramsch et al. (2019) we show the application of a deep multi-layer perceptron for map-based 4D seismic pressure saturation inversion. In this work we show the information gain of feed-forward multi-layer perceptron neural networks by including an explicit calculation of the AVO gradient within the network architecture. It's exemplary for including domain knowledge as a prior in machine learning.

Method

We build a deep feed-forward network to invert seismic amplitude maps for pressure and saturation changes. We use the high-level Python framework `keras` with a `tensorflow` backend. The neural network was trained on synthetic data, to subsequently predict field data. The network takes the seismic input samplewise with near, mid, and far stacks, and pore volume. We inject 20% Gaussian noise to model the noisier field data directly after the input layer. This is fed to a custom layer that calculates the PP AVO gradient between far-mid, mid-near, and far-near. The main components are as follows:

Gaussian noise injection

The synthetic model is noise-free. While we get good results on the training data and the modelled test data, the network does not transfer well to noisy field data. Although the 4D NRMS is very low in the data set, the sample-wise fluctuations in the field seismic differ significantly from the synthetic data. We apply additive Gaussian noise with $\sigma = .02$ to the seismic inputs separately to simulate independent fluctuations of the seismic maps. This significantly decreases the training and validation performance on noise free synthetic data. On field data, however, this enables good transfer of the neural network.

```
noisy_input = GaussianNoise(0.02)(input_data)
```

Explicit AVO gradient calculation

The Schiehallion field is a good example of imbalanced learning. We have many samples of pressure changes ΔP, a good selection of water saturation changes ΔS_w, and very few gas saturation changes ΔS_g. Yet, the changes in gas saturation ΔS_g produce the strongest changes in seismic P wave amplitudes. Statistically, these can easily be regarded as outliers, and therefore, possibly disregarded by the neural network. From decades of seismic analysis, we know that the AVO gradient is very good for pressure saturation separation. We implement an explicit calculation of AVO gradients in the network.

$$G = \frac{A_{\Theta_1} - A_{\Theta_0}}{x_{\Theta_1} - x_{\Theta_0}}, \tag{1}$$

where G is the PP AVO gradient, A is the seismic P wave amplitude, x is the offset, and Θ is the angle.

```
mid_near = Lambda(
    lambda inputs: (inputs[0] - inputs[1]) / (10)
)([noisy_mid, noisy_near])

far_mid = Lambda(
    lambda inputs: (inputs[0] - inputs[1]) / (10)
)([noisy_far, noisy_mid])

far_near = Lambda(
    lambda inputs: (inputs[0] - inputs[1]) / (20)
)([noisy_far, noisy_near])
```

Encoder-decoder architecture

Subsequently, the four input maps and the three gradient maps are concatenated and fed to an encoder architecture that condenses the information to an embedding layer z. This layer learns a collection of Gaussian distributions to represent the noisy input data The decoder samples this variational embedding layer to calculate the pressure change ΔP, change in water saturation ΔS_w, and gas saturation ΔS_g.

The full architecture is of the encoder-decoder class. The encoder reduces the number of parameters with each subsequent layer. This forces the network to learn a lossy compression of the input data as z-vector. The decoder increases the number of nodes per layer toward the output. The network therefore learns to correlate the low resolution representation with the desired output.

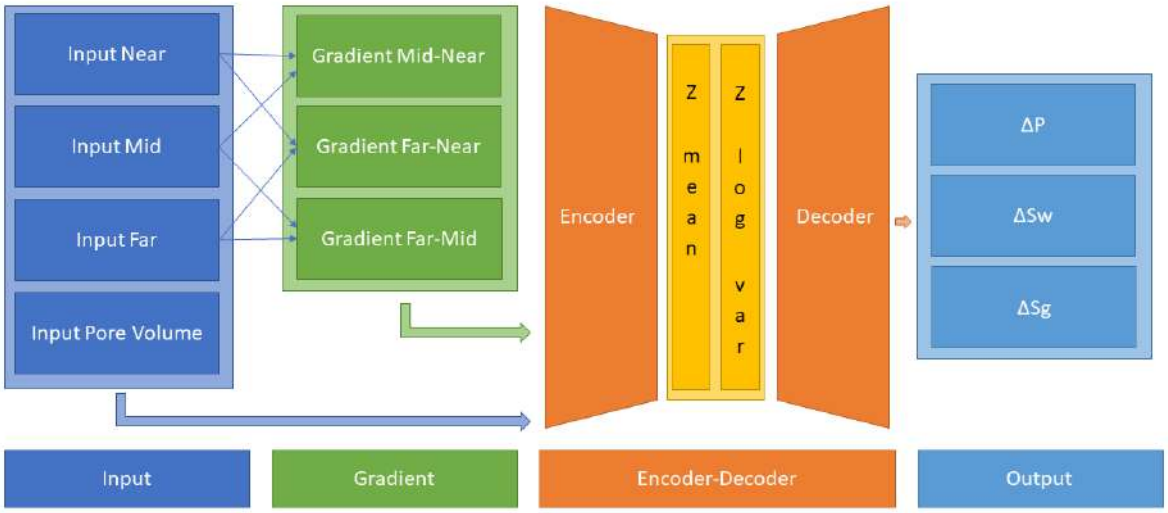


Figure 1 *Full Architecture from Dramsch et al. (2019).*

Variational Z Vector

The inversion of noisy input benefits from a variational representation of compressed z-vector. The networks learns Gaussian distributions in the embedding layer. Therefore, we have to apply the reparametrization trick outlined in Kingma and Welling (2013) to circumvent the sampling process cannot be learned by gradient descent. We use the implementation in Chollet (2015) for variational autoencoders.

Results

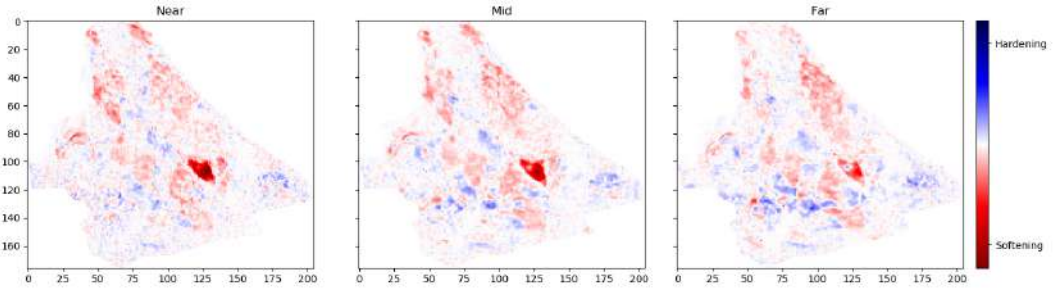


Figure 2 *Schiehallion 2004 Timestep Seismic data, pore volume and sim2seis results.*

In figure 2 we show the 2004 time step of the Schiehallion 4D. Figure 3 contains the inversion result using the variational encoder decoder architecture. Some coherency in the maps can be seen, but each map is very noisy and the gas saturation map contains many data points that indicate gas desaturation, which cannot be confirmed by production data.

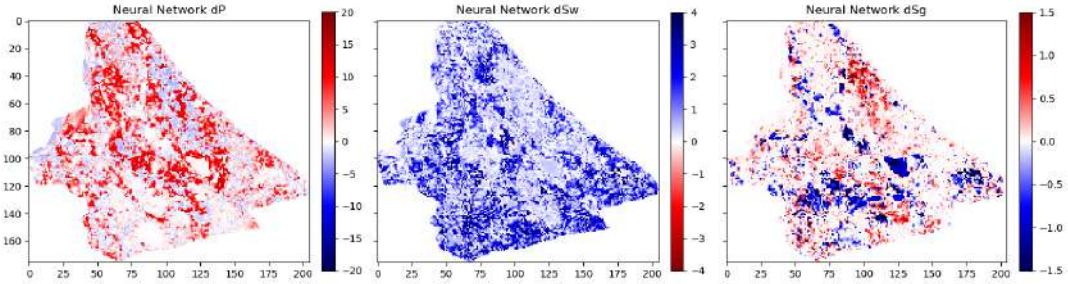


Figure 3 *Variational Encoder Decoder Architecture Inversion*

When we add the gradient, we can clean up some of the misfit in the gas saturation maps ΔS_g. Particularly, the event with the strongest softening in the amplitude maps, is partially reassigned to the pressure map ΔP. However, the inversion process is still very prone to noise. In figure 5, we show the inversion results of a AVO-gradient neural network with a noise injection at training of $\sigma = .02$. The inversion maps are very coherent. Noise injection without gradient calculation does not give adequate results.

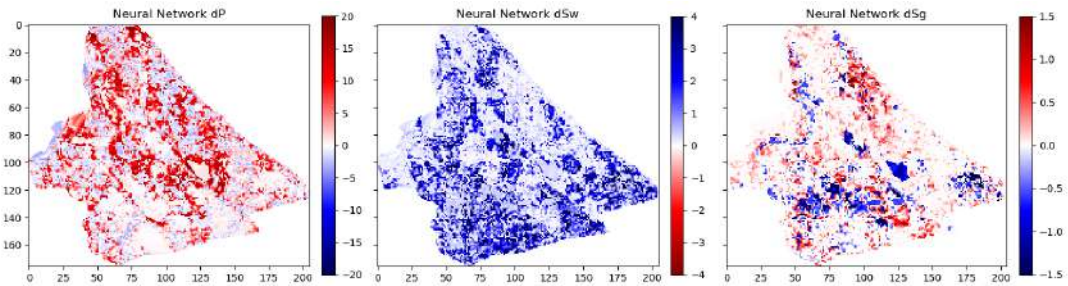


Figure 4 *AVO-Gradient Variational Encoder Decoder Architecture Inversion*

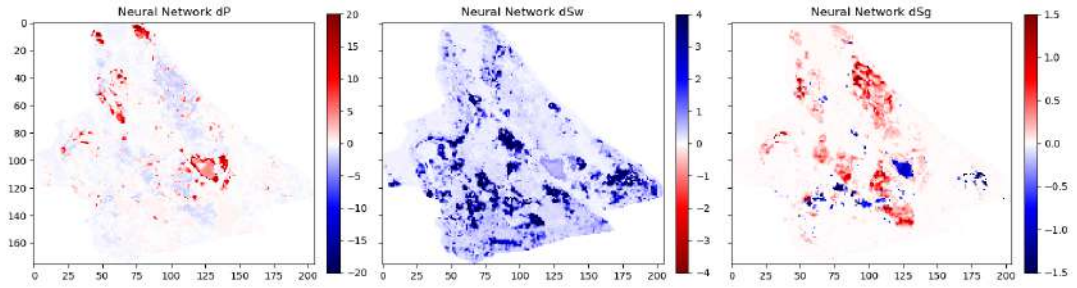


Figure 5 *Noiseinjected AVO-Gradient Variational Encoder Decoder Architecture Inversion*

Conclusions

We have shown a neural network architecture that incorporates physical domain knowledge to enable transfer from synthetic to field data. The final inversion result has very good coherency, despite the network not having any spatial context. While further investigation is necessary, this indicates that useful information has been learned. This is one example, where bias can be intentionally introduced into the network architecture to include physics into machine learning.

Acknowledgements

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. We thank the sponsors of the Edinburgh Time-Lapse Project, Phase VII (AkerBP, BP, CGG, Chevron, ConocoPhillips, ENI, Equinor, ExxonMobil, Halliburton, Nexen, Norsar, OMV, Petrobras, Shell, Taqa, and Woodside) for supporting this research. The Brazilian governmental research-funding agency CNPq. We are also grateful to Linda Hodgson and Ross Walder for important discussions on the field and dataset.

References

Chollet, F. [2015] Keras. https://github.com/fchollet/keras.

Dramsch, J.S., Corte, G., Amini, H., Lüthje, M. and MacBeth, C. [2019] Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data.

Dramsch, J.S. and Lüthje, M. [2018] Deep-learning seismic facies on state-of-the-art CNN architectures. In: *SEG Technical Program Expanded Abstracts 2018*, Society of Exploration Geophysicists.

Grün, F., Rupprecht, C., Navab, N. and Tombari, F. [2016] A taxonomy and library for visualizing learned features in convolutional neural networks. *arXiv preprint arXiv:1606.07757*.

He, K., Zhang, X., Ren, S. and Sun, J. [2015] Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: *Proceedings of the IEEE international conference on computer vision*. 1026–1034.

Ioffe, S. and Szegedy, C. [2015] Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Karpatne, A., Watkins, W., Read, J. and Kumar, V. [2017] Physics-guided neural networks (pgnn): An application in lake temperature modeling. *arXiv preprint arXiv:1710.11431*.

Kingma, D.P. and Welling, M. [2013] Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

de Oliveira, L., Paganini, M. and Nachman, B. [2017] Learning particle physics by example: location-aware generative adversarial networks for physics synthesis. *Computing and Software for Big Science*.

Roeth, G. and Tarantola, A. [1994] Neural networks and inversion of seismic data. *Journal of Geophysical Research: Solid Earth*, **99**(B4), 6753–6768.

Schütt, K.T., Arbabzadah, F., Chmiela, S., Müller, K.R. and Tkatchenko, A. [2017] Quantum-chemical insights from deep tensor neural networks. *Nature communications*, **8**, 13890.

Ulyanov, D., Vedaldi, A. and Lempitsky, V. [2018] Deep image prior. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9446–9454.

6.2 Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data

Abstract: In this work we present a deep neural network inversion on map-based 4D seismic data for pressure and saturation. We present a novel neural network architecture that trains on synthetic data and provides insights into observed field seismic. The network explicitly includes AVO gradient calculation within the network as physical knowledge to stabilize pressure and saturation changes separation. We apply the method to Schiehallion field data and go on to compare the results to Bayesian inversion results. Despite not using convolutional neural networks for spatial information, we produce maps with good signal to noise ratio and coherency.

Key points:

- Physics-based Deep Neural Network
- AVO gradient calculation in network architecture stabilizes prediction
- Variational bottleneck to buffer noisy inputs
- Noise injection at input to transfer from simulation to field data

J. S. Dramsch, G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019d). "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data". In: *Second EAGE Workshop Practical Reservoir Monitoring 2019*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201900028

EAGE

Introduction

Estimating reservoir property change during a period of production from 4D seismic data has been a concentrated challenge and ambition for geoscientists in the oil and gas industry. These estimates can contribute to a better history matching of the reservoir simulation and for more comprehensive reservoir monitoring.

With the advance of machine learning techniques on all fronts in the geosciences we can address what roles machine learning can take in the established pressure and saturation inversion workflows and what other new workflows can be constructed using this tool. Machine learning is such a broad concept that it can be incorporated at different levels on all the current well established workflows to diminish their weaknesses, bringing more value to the pressure and saturation estimations from seismic inversion. Not only that, with this tool we can create completely new workflows that we are only beginning to grasp.

Here we will present results for two separate methodologies of seismic inversion to changes in pressure and saturation. The first method is a well established model-based Bayesian inversion method using a calibrated petro-elastic model and convolution workflow as the forward seismic modeling operator. In the second method we use a deep neural network to model the inversion process, we use synthetic seismic data to train the network, then apply the inversion to observed data. The methods are applied to the same field data giving a nice platform to compare the neural network inversion results to a more conventional approach.

Schiehallion Data

The inversions are applied to maps of Schiehallion's upper T31 sandstone. It is a fairly thin reservoir (5-30m), which is well defined in the seismic data by one single trough. For this reason, a map-based approach is appropriate. Schiehallion is a highly compartmentalized field with initial pressure close to bubble point pressure. Production in this complex structure led to areas with strong pressurization due to water injection into closed compartments, while other areas lack the pressure support and experience gas release due to pressure depletion. We face the challenge of inverting 4D seismic data to changes in pressure, water saturation and gas saturation (ΔP, ΔSw and ΔSg), so the methods need to deal properly with the non-linearities due to each of these effects. The seismic data analysed is a set of eight vintages (from 1996 to 2010). These were reprocessed by CGG in 2014, following a 4D driven multi-vintage workflow. The processing workflow was carefully optimized to maintain 4D AVO amplitudes intact. Synthetic feasibility studies showed that the 4D AVO attributes are in line with the theoretical expectations. The seismic data used for inversion is the 4D difference of the sum of negative amplitudes (ΔSNA) map attribute, extracted from three angle-stacks, along the reservoir time window (see figure 2).

Method 1 - Model-based Bayesian inversion

The Bayesian invesion workflow is explained in detail in Corte et al. (submitted 2019). Essentially the workflow uses a petro-elastic model calibrated to the seismic data by Amini (2018) and a convolutional step to model the seismic data. The ΔSNA attribute is then extracted from the synthetic seismic and compared to the real seismic ΔSNA map. Since this is a map-based inversion, all realizations are sampled in map form and then go through a conversion into the vertical reservoir simulation grid in order to run the forward modelling process. We use a monte carlo sampling algorithm to generate thousands of realizations of the full map and from these extract best estimations and uncertainties. This inversion is constructed in a Bayesian model-based form, with the objective of bringing together information from the history matched reservoir simulation and seismic data. Reservoir simulation results for ΔP, ΔSw and ΔSg are incorporated as prior knowledge, to settle ambiguities and lack of seismic information. Where the seismic data lacks information about a certain property the method will bring this information from the simulation model. The inversion results will deviate from the simulation in areas where the seismic data contains enough consistent information to indicate an update is necessary.

Method 2 - Neural network inversion

We use a deep neural network to model the inversion process, based on the synthetic convolution seismic data. Although convolutional neural networks are considered the state of the art in spatially correlated data, we show that a sample-wise feed forward neural network trained on noise-free convolutional seismic can invert observed seismic data. We aim to build a regression model that can invert physical seismic angle stack data to pressure and saturation data.

EAGE

Distinguishing pressure and saturation changes in 4D seismic data is a hard to solve problem. In neural networks, this is no different. The variation of data showing different pressure and saturation change scenarios is sparse, which complicates training and may possibly be disregarded as noise. This increases the need for training data immensely. However, we can include prior physical insights into neural networks to reduce the cost of training and uncertainty. As neural networks are at its basis very large mathematical functions, we can explicitly calculate the P-wave AVO gradient within the network to use as additional information source, without the need of feeding it into the network as input data. This has the added benefit of the network learning on noisy gradients. The design choice for the neural networks can be arbitrary, however, encoder-decoder networks have proven to force neural networks to find meaningful relationships within the data and reduce to these in the bottleneck or embedding layer. For the final architecture we used `hyperopt` (Bergstra et al., 2013) and `keras` (Chollet, 2015). This allows us to use a Tree of Parzen (TPE) estimator for hyperparameter estimation. The estimator models $P(x|y)$ and $P(y)$, where y the quality of fit and x is the hyperparameter set drawn from a non-parametric density (Bergstra et al., 2011).

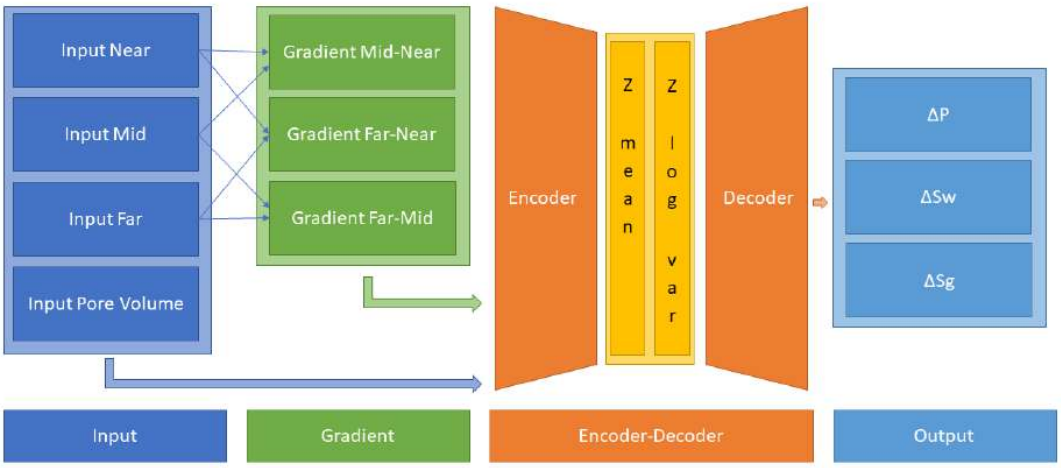


Figure 1 *Architecture for sample-based seismic inversion with explicit gradient calculation.*

The architecture is shown in figure 1. Inputs are Near, Mid, Far seismic, and Pore volume. These Input Layers are passed on to calculate the mid-near, far-mid, and far-near gradients. These four inputs and three gradients are concatenated and fed to the encoder. z_mean and z_log_var build the variational embedding with z_Lambda being the sampler fed to the decoder network. The decoder splits into three output layers ΔP, ΔSw, and ΔSg.

The network is trained using sim2seis results calculated for the seven time-steps at seismic monitor acquisition times, it is then used to invert each seismic monitor individually. The inversion results for the synthetic data gave a consistent R^2-score of over 0.6 for all simultaneous inversion targets ΔP, ΔSw and ΔSg with an encoder-decoder architecture and a deterministic embedding layer. While we kept the main architecture constant, we replaced the embedding layer with a variational formulation to allow for noise in the input to output mapping added noise injection to the input layer, to apply Gaussian Noise during the training phase. This significantly improved the inference on observed seismic data. The total training time for the network was 3 hours on a K5200 GPU, prediction speed takes 5.11 $s \pm 22.1$ ms.

Schiehallion Field Data Example

The field data differs significantly from the synthetic data in that it is noisier, assuming the same ground truth. This is a true challenge for a sample-wise process to produce consistent results. We have trained the network with Gaussian noise on the input data with zero mean and a standard deviation of $\sigma = .02$, therefore, approximately 95 % of the noise may distort up to a maximum 40 % of the clean signal.

Figure 2 shows the observed 4D seismic maps (ΔSNA) for the 2004 monitor acquisition using the 1996 acquisition as baseline. Figure 3 shows, in the first row, the simulation model results (used in the Bayesian method as prior information), in the second row, the inversion results for the Bayesian method, and in the third row, the inversion results for the neural network method.

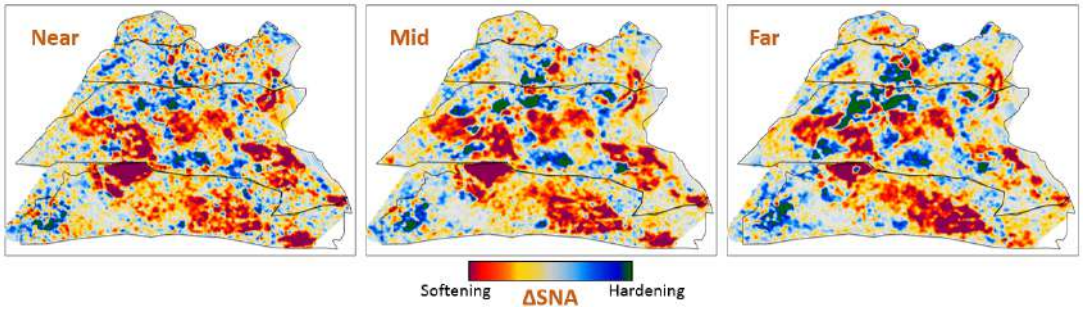


Figure 2 *Schiehallion 2004 Timestep Seismic data, pore volume and sim2seis results.*

From figure 3 we can see clearly the influence of the prior simulation model in the Bayesian results. The neural network does not use a prior, so the results are not influenced by the simulation model and can be seen as a direct interpretation of the seismic data. Comparing both we can see what bits of information the Bayesian method is bringing from the prior. The seismic data is most sensitive to gas saturation changes, so the Bayesian method is able to capture this consistent information from seismic data and deviate ΔSg results from the initial prior. The results for gas saturation are the most in agreement in both methods precisely because all this information is coming from the seismic data. We see some leakage of hardening effects into the ΔSg results in method 2 due to the fact that we cannot set constraints to that inversion process. Since there is no initial gas saturation in those areas the saturation change cannot be negative, these comprehensive constraints are imbedded into the Bayesian workflow but not in the neural network.

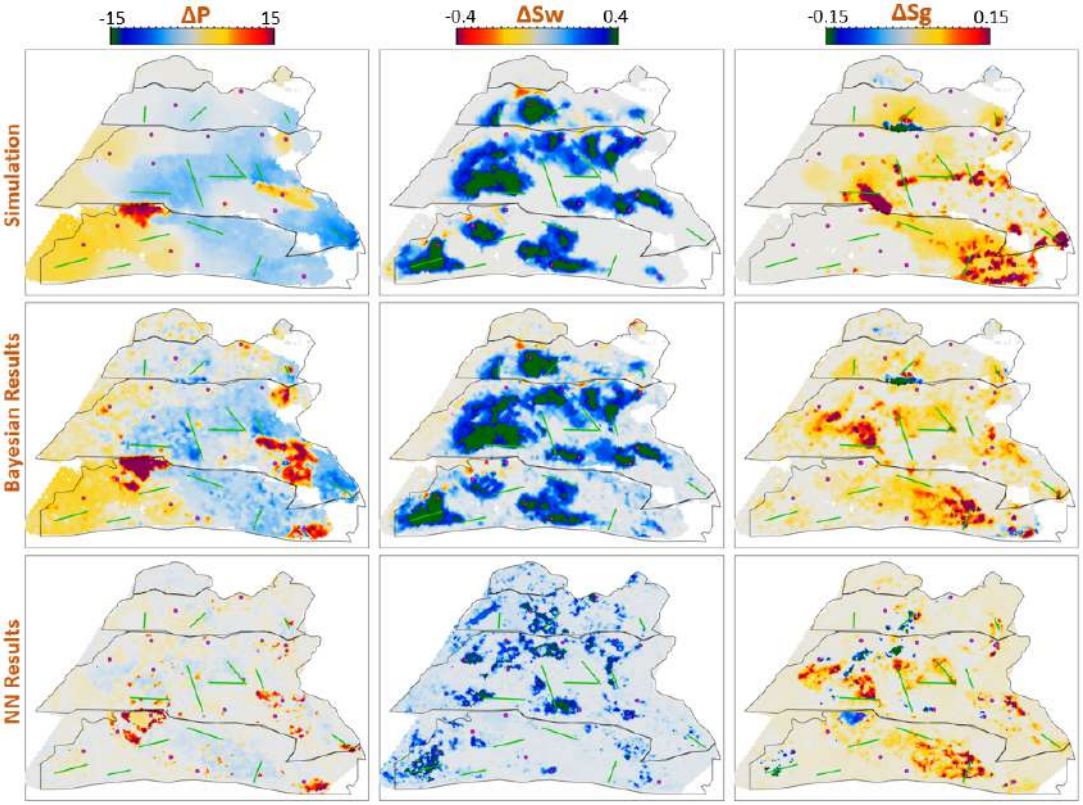


Figure 3 *Schiehallion 2004 Timestep Bayesian Inversion and Neural Inversion*

Water saturation has a distinctive hardening effect on seismic data, but in this map it is highly obscured by stronger overlying softening effects due to pressure increase and gas breakout. The neural network interprets all the hardening anomalies correctly as water saturation increase, while controlling for noise in areas of softening amplitudes. In those areas the seismic data does not contain useful information on the water saturation so the Bayesian result relies on a strong prior to compensate. All of the water saturation inverted by method 2 is in agreement with method 1, but since method 1 has this additional information from the prior, the map seems more coherent.

The pressure effect on seismic is highly non-linear. While high increases in pressure show a very strong softening effect, milder pressure variations (up to ± 7 *MPa*) have very little influence on the seismic data and are easily obscured by overlying effects. For this reason, the neural network pressure inversion in regions of mild change is low and often correlated with saturation. The Bayesian inversion benefits from the prior to fill those pressure values. This method does deviate from the prior in areas of strong softening signals due to pressure increase, and those areas are also correctly interpreted by the neural network inversion.

When we relax the prior of the Bayesian inversion, these results are very noisy in the pressure and water saturation estimates. In these areas the neural network inversion is robust to noise. During the neural network training the pore volume has shown to be important in guiding the inversion from the seismic data. Adding pore volume data adds a structural component to the neural inversion process, which improves the overall results from the sample-based method significantly.

Conclusions

This work presents Deep Neural Inversion of 4D seismic data. We compare the results with a Bayesian Inversion approach. We show that Deep Neural Networks can model seismic inversion trained on synthetic data. Explicit calculation of the P-wave AVO gradient within the network stabilizes the pressure-saturation separation within the network and Noise Injection enables the transfer to unseen seismic field data. Neural networks can be an important tool to investigate nascent information in 4D seismic data to improve inversion workflows and reduce uncertainty in seismic analysis.

The Neural Inversion can be used as a valuable tool to explore purely data-based inversion results in the presence of noise. It is able to translate the ambiguous seismic amplitudes into much more easily interpreted property maps. The value of the Bayesian inversion results presented is in combining all knowledge about the reservoir to create a general view of the reservoir dynamics. These results show the current understanding of reservoir dynamics updated by imprinting seismic information on top of the history matched simulation results.

Acknowledgements

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. We thank the sponsors of the Edinburgh Time-Lapse Project, Phase VII (AkerBP, BP, CGG, Chevron, ConocoPhillips, ENI, Equinor, ExxonMobil, Halliburton, Nexen, Norsar, OMV, Petrobras, Shell, Taqa, and Woodside) for supporting this research. The Brazilian governmental research-funding agency CNPq. We are also grateful to Linda Hodgson and Ross Walder for important discussions on the field and dataset. We thank Mikael Lüthje for valuable feedback.

References

Amini, H. [2018] Comparison of Xu-White, Simplified Xu-White (Keys & Xu) and Nur's Critical Porosity in Shaley Sands. In: *80th EAGE Conference and Exhibition 2018*.

Bergstra, J., Yamins, D. and Cox, D.D. [2013] Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures.

Bergstra, J.S., Bardenet, R., Bengio, Y. and Kégl, B. [2011] Algorithms for Hyper-Parameter Optimization. In: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F. and Weinberger, K.Q. (Eds.) *Advances in Neural Information Processing Systems 24*, Curran Associates, Inc., 2546–2554.

Chollet, F. [2015] Keras. https://github.com/fchollet/keras.

Corte, G., MacBeth, C. and Amini, H. [submitted 2019] North Sea field application of 4D Bayesian in-

version to pressure and saturation changes. In: *81st EAGE Conference & Exhibition 2019*. Submitted.

CHAPTER 7

Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks

Abstract: We present a novel 3D warping technique for the estimation of 4D seismic time-shift. This unsupervised method provides a diffeomorphic 3D time shift field that includes uncertainties, therefore it does not need prior time-shift data to be trained. This results in a widely applicable method in time-lapse seismic data analysis. We explore the generalization of the method to unseen data both in the same geological setting and in a different field, where the generalization error stays constant and within an acceptable range across test cases. We further explore upsampling of the warp field from a smaller network to decrease computational cost and see some deterioration of the warp field quality as a result.

Key points:

- 3D time shift extraction
- Diffeomorphic constraint models geological intuition
- Unsupervised training does not need prior time shifts
- Generalizes to same field with different acquisition
- Generalizes to different field with differen geological setting

J. S. Dramsch, A. N. Christensen, C. MacBeth, and M. Lüthje (2019b). "Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks". In: *IEEE Transactions in Geoscience and Remote Sensing*. Submitted, Chapter 7

Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks

Jesper Sören Dramsch, Anders Nymark Christensen, Colin MacBeth, Mikael Lüthje

***Abstract*—We present a novel 3D warping technique for the estimation of 4D seismic time-shift. This unsupervised method provides a diffeomorphic 3D time shift field that includes uncertainties, therefore it does not need prior time-shift data to be trained. This results in a widely applicable method in time-lapse seismic data analysis. We explore the generalization of the method to unseen data both in the same geological setting and in a different field, where the generalization error stays constant and within an acceptable range across test cases. We further explore upsampling of the warp field from a smaller network to decrease computational cost and see some deterioration of the warp field quality as a result.**

***Index Terms*—4D seismic, time-lapse, deep learning, unsupervised learning, 3D time-shift, neural network**

I. Introduction

SEISMIC time-lapse data consists of two 3D reflection amplitude cubes that represent the subsurface they were collected from. These cubes are acquired years apart with expected changes in the subsurface due to e.g. hydrocarbon production. The differences in the subsurface cause changes in both amplitudes and velocities, which introduces misalignment of seismic reflectors. Measuring the misalignment and aligning these surfaces to obtain a reliable difference cube is one of the main disciplines in 4D seismic processing.

These time shifts are most commonly obtained by windowed cross-correlation and other statistical or signal processing approaches [1]. Considering the recent advances of machine learning in imaging and domain transfer, we explore possibilities of alignment with convolutional neural networks. Machine learning approaches, however, most commonly require labeled data to find a mapping $f(x) = y$, with x being the input data, f being the blackbox algorithm like a neural network, and y being the labels or target.

A common problem in machine learning for subsurface science is determining the ground truth. Obtaining information from the subsurface is often prohibited by cost, and e.g. core samples are highly localised data that is often altered by the extraction method as well as the sheer act of unearthing the sample. Additionally, synthetic data may introduce the inverse crime [2] of using the same theory to generate and invert data. Luckily, the physics of medical imaging and inversion is very similar to geophysics, where methods can be validated and fine-tuned. The main method discussed in this paper is adapted from the medical imaging literature.

The lack of ground truths leads to another problem that deep learning address but do not solve. For classic neural networks, we need to know a target label dataset, i.e. knowing a prior warp velocity. In 4D seismic this would mean employing an established method to obtain time shifts. This would effectively result in abstracting that method in a neural network, or modelling the warp, which would lead to committing the inverse crime. Logically, this lead us to explore unsupervised methods.

We discuss several options for architectures for mapping the monitor seismic cube to the base seismic cube directly within the network. This is possible in unsupervised configurations but depending on the architecture of the network this problem can be ill-constrained and generate non-physical mappings. One warranted criticism of deep learning and neural networks is the lack of explainability and limited interpretability. However, we employ a deep neural network to obtain warp velocity vectors, a 3D equivalent of time shifts, for dense deterministic warping instead of directly obtaining the warped result from a neural network. This enables us to interpret the warping vectors and constrain the warp path in addition to the warp result.

Moreover, we present the first 4D seismic 3D time shift estimator with uncertainty measures. We achieve this by implementing a variational layer that samples from a Gaussian with the reparametrization trick [3]. Therefore, we can counteract some of the influence of noise on the performance of the network.

II. Theory

Extracting time shifts from 4D seismic data is most commonly done trace-wise (1D), which limits the problem to depth. This provides sufficient results for simple problems. However, geologically complex systems and pre-stack time shifts benefit from obtaining 3D time-shifts. We discuss classical 3D time-shift extraction methods, we then go on to discuss relevant deep learning methods. These methods extract time-shifts with different constraints which we explore. For brevity we present the results of the best method to date, developed for the medical domain: VoxelMorph [4].

The goal of both conventional and machine learning methods is to obtain a warp velocity field $\mathbf{u}(x, y, z)$ that ideally aligns two 3D cubes B and M within given constraints. That means a sample $m[x, y, z]$ will be aligned by adjusting $m[x+u_x, y+u_y, z+u_z]$. In image processing this is considered "dense alignment" or "dense warping", hence we need a dense vector field to align each sample in the base and the monitor cube. Generally, $\mathbf{u}(x, y, z) \in \mathbb{R}^3$, which implies interpolation to obtain the warped result.

A. Conventional Methods

Most conventional methods in 4D seismic warping focus on 1D methods [5], which include local 1D cross-correlation, dynamic time warping [6], optical flow methods and methods based on Taylor expansion [7]. We do not cover these methods in detail, but focus on the limited applications of 3D methods in 4D seismic warping.

1) Local 3D Cross Correlation: Hall et al [8] introduced local 3D cross-correlation as a method for surface-based image alignment. The horizon-based nodal cross-correlation results were then linearly interpolated to full cubes. Hale et al [9] extended this method to full seismic cubes by calculating the multi-dimensional cross-correlation windowed by a Gaussian with a specified radius. The correlation results are normalized to avoid spurious correlations by amplitude fluctuations and high-amplitude events. Subsequently the cross-correlation result is searched for peaks using the following triple sum:

$$c[u_x, u_y, u_z] = \sum_{x,y,z=-\infty}^{\infty} b[x, y, z] \cdot m[x + u_x, y + u_y, z + u_z], \tag{1}$$

with c being the cross-correlation lag. The computational complexity of this method is $\mathcal{O}(N_s \times N_l)$ with N_s being the total number of samples and N_l being the total number of lags.

Stabilization of the results of 3D cross-correlation is obtained by applying spectral whitening of the signals and smoothing the images with a Gaussian filter without increasing the computational complexity despite the windowing function [9].

2) Inversion-based methods: Rickett et al [10] describe a non-linear inversion approach, with the objective function being

$$\mathbb{E} = |\mathbf{d} - f(\mathbf{m})|^2 + |\nabla_x(\mathbf{m})|^2 + |\nabla_y(\mathbf{m})|^2 + |\nabla_z^2(\mathbf{m})|^2 \tag{2}$$

with $\mathbf{m}$ being the model vector, $\mathbf{d}$ being the data vector. The non-linear inversion is constrained by applying the first-derivative to the spatial dimensions z, y and Laplacian in z to obtain a smooth solution. Cherrett et al [11] implement a geostatistical joint inversion that uses the geostatistical information combined with data constraints as a prior in a Bayesian inversion scheme.

$$P(x|geostats, data) \propto \exp\left(-(\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}}\mathbf{C}^{-1}(\mathbf{x} - \boldsymbol{\mu})/2\right) \tag{3}$$

with $\mathbf{C}$ being the posterior covariance matrix, $\mathbf{x}$ the sample mean vector and $\boldsymbol{\mu}$ being the posterior mean vector.

B. Machine Learning Methods

The machine learning methods discussed in this section are imaging based, and therefore rely on recent advances of convolutional neural networks (CNN) in deep learning. We discuss different approaches that include supervised and unsupervised / self-supervised methods. These methods are all based on convolutional neural networks (CNNs).

CNNs are a type of neural network that is particularly suited to imaging approaches. They learn arbitrary data-dependent filters that are optimized based on the chosen objective via gradient descent. These filters can operate on real images, medical images, or seismic data alike. The convolutional filter benefits from weight sharing, making the operation efficient and particularly suited to GPUs or specialized hardware. In Figure 1 we show a schematic image, that is convolved with moving 3x3 filters repeatedly to obtain a spatially down-sampled representation. These convolutional layers in neural networks can be arranged in different architectures that we explore in the following analysis of prior methods in image alignment.

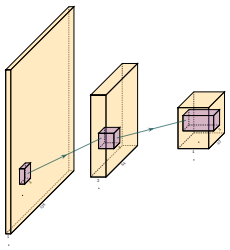


Fig. 1. Schematic convolutional neural network. The input layer (yellow) is convolved with a 3×3 filter that results in a spatially subsampled subsequent layer that contains the filter responses. This second layer is again convolved with a 3×3 filter to obtain the next layer. Subsampling is achieved by strided convolutions or pooling.

1) Supervised CNNs: Supervised end-to-end CNNs rely on reliable ground truth, including the time shifts being available. Training a supervised machine learning system requires both a data vector x and a target vector y to train the blackbox system $f(x) \Rightarrow y$. This means that we have to provide extracted time-shifts from other methods, which implicitly introduce assumptions from that method into the supervised model. Alternatively, expensive synthetic models would be required.

The supervised methods are largely based on Optical Flow methods [12], [13]. The FlowNet [12] architecture is based on an Encoder-Decoder CNN architecture. Particularly, FlowNet has reached wide reception and several modifications were implemented, namely FlowNet 2.0 [14] improving accuracy, and LiteFlowNet [15] reducing computational cost. SpyNet [13] and PWC-Net [16] implement stacked coarse-to-fine networks for residual flow correction. PatchBatch [17] and deep discrete flow [18] implement Siamese Networks [19] to estimate optical flow. Alternatively, DeepFlow [20] attempts to extract large displacements optical flow using pyramids of SIFT features. These methods introduce varying types of network architectures, optimizations, and losses that attempt to solve the optical flow problem in computer vision.

2) Unsupervised CNNs: Unsupervised or self-supervised CNNs only rely on the data, relaxing the necessity for ground truth time shifts. In [21] the FlowNet architecture is reformulated into an unsupervised optical flow estimator with bidirectional census loss called UnFlow. The UnFlow network relies on the smooth estimation of the forward and backward loss, then adds a consistency loss between the forward and backward loss and finally warps the monitor to the base image to obtain the final data loss. Optical flow has historically underperformed on seismic data, due to both smoothness

and illumination constraints. However, UnFlow replaces the commonly used illumination loss by a ternary census loss [22] with the ϵ-modification by [23]. While this bears possible promise for seismic data, UnFlow implements 2D losses as opposed to a 3D implementation that we focus on.

3) Cycle-consistent Generative Adversarial Networks: Cycle-GANs are a unsupervised implementation of Generative Adversarial Networks that are known for domain adaptation [24]. These implement two GAN networks that perform a forward and backward operation that implements a cycle-consistent loss in addition to the GAN loss. The warping problem can be reformulated as a domain adaptation problem. This implements two Generator networks F and G and the according discriminators D_X and D_Y. These perform a mapping $G : X \rightarrow Y$ and $F : Y \rightarrow X$, trained via the GAN discrimination. The cycle-consistency implements $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$ with the backwards cycle-consistency being $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$.

Cycle-GANs such as pix2pix [25] separate image data into a content vector and a texture vector, which could bear promise in the seismic domain, adapting a wavelet vector and an interval vector [26]. However, the confounding of imaging effects, changing underlying geology, changing acquisition, etc makes the separation non-unique. Moreover, extracting the time shift information and conditioning in the GAN is a very complex problem. The Recycle-GAN [27] addresses temporal continuity in videos, this is however hard to transfer to seismic data, considering the low number of time-steps in a 4D seismic survey as opposed to videos. Furthermore, the lack of interpretability of GANs at the point of writing, prohibits GANs from replacing many physics-based approaches, like the extraction of time-shifts.

III. Method

The Voxelmorph [4] implements a U-net [28] architecture to obtain a dense warp velocity field and subsequently warps the monitor cube to match the base cube. This minimizes assumptions that have to be satisfied for applying optical flow-based methods. Additionally, the Voxelmorph architecture was specifically developed on medical data. Medical data often has few samples, like seismic data, as opposed to popular video datasets, which FlowNet and derivative architectures are geared towards application of popular video datasets. A U-net architecture is particularly suited for segmentation tasks and transformations with smaller than usual amounts of data, considering it was introduced on a small biomedical dataset. The short-cut concatenation between the input and output layers stabilizes training and avoids the vanishing gradient problem. It is particularly suited to stable training in this image matching architecture. In Figure 2 the U-Net is the left-most stack of layers, aranged in an hourglass architecture with shortcuts. These feed into a variational layer $\mathcal{N}(\mu, \sigma)$, the variational layer is sampled with the reparametrization trick, due to the sampler not being differentiable [3]. The resulting differential flow is integrated using the VecInt layer, which uses Scaling and Squaring [29]. Subsequently, the data is passed into a spatial transformation layer. This layer transforms the monitor cube according to the warp velocity field obtained from the integrated sampler. The result is used to calculate the data loss between the warped image and the base cube.

More formally, we define two 3D images $\boldsymbol{b}, \boldsymbol{m}$ being the base and monitor seismic respectively. We try to find a deformation field ϕ parameterized by the latent variable z such that $\phi_z : \mathbb{R}^3 \rightarrow \mathbb{R}^3$. The deformation field itself is defind by this ordinary differential equation (ODE):

$$\frac{\partial \phi^{(t)}}{\partial t} = v(\phi^{(t)}), \tag{4}$$

where t is time, v is the stationary velocity and the following holds true $\phi^{(0)} = \boldsymbol{I}$. The integration of v over $t = [0, 1]$ provides $\phi^{(1)}$. This integration represents and implements the one-parameter diffeomorphism in this network architecture. The variational Voxelmorph formulation assumes an approximate posterior probability $q_\psi(z|\boldsymbol{b}; \boldsymbol{m})$, with ψ representing the parameterization. This posterior is modeled as a multivariate normal distribution with the covariance $\Sigma_{z|m,b}$ being diagonal:

$$q_\psi(z|\boldsymbol{b}; \boldsymbol{m}) = \mathcal{N}(z, \boldsymbol{\mu}_{z|m,b}, \Sigma_{z|m,b}), \tag{5}$$

the effects of this assumption are explored in [30].

The approximate posterior probability q_ψ is used to obtain the variational lower bound of the model evidence by minimizing the Kullback-Leibler (KL) divergence with $p(z|\boldsymbol{b}; \boldsymbol{m})$ being the intractable posterior probability. Following the full derivation in [30], considering the sampling of $z_k \sim q_\psi(z|\boldsymbol{b}, \boldsymbol{m})$ for each image pair $(\boldsymbol{b}, \boldsymbol{m})$, we compute $\boldsymbol{m} \circ \phi_{z_k}$ the warped image we obtain the loss:

$$\begin{aligned}
\mathcal{L}(\psi; \boldsymbol{b}, \boldsymbol{m}) &= -\mathbf{E}_q[\log p(\boldsymbol{b}|z; \boldsymbol{m})] \\
&\quad + \mathbf{KL}[q_\psi(z|\boldsymbol{b}; \boldsymbol{m})||p_\psi(z|\boldsymbol{b}; \boldsymbol{m})] \\
&\quad + \text{const} \\
&= \frac{1}{2\sigma^2 K} \sum_k ||\boldsymbol{b} - \boldsymbol{m} \circ \phi_{z_k}||^2 \\
&\quad + \frac{1}{2}[\mathbf{tr}(\lambda \boldsymbol{D} \Sigma_{z|x;y}) - \log \Sigma_{z|x;y}) \\
&\qquad + \boldsymbol{\mu}_{z|m,b}^T \boldsymbol{\Lambda}_z \boldsymbol{\mu}_{z|m,b}] + \text{const},
\end{aligned} \tag{6}$$

where Λ_z is a precision matrix, enforcing smoothness by the relationship $\Sigma_z^{-1} = \Lambda_z = \lambda \boldsymbol{L}$, λ controlling the scale of the velocity field. $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ is the Laplacian of a neighbourhood graph over the voxel grid, where $\boldsymbol{D}$ is the graph degree matrix, and A defining the voxel neighbourhood. K signifies the number of samples. We can sample $\boldsymbol{\mu}_{z|m,b}$ and $\Sigma_{z|m,b}$ as variational layers in a neural network . Given the diagonal constraint on Σ, we define the variational layer as the according standard deviation σ of the corresponding dimension. Therefore we sample $\mathcal{X} \sim \mathcal{N}(\mu, \sigma^2)$ using the reparameterization trick first implemented in variational auto-encoders [31]

Defining the architecture and losses as presented in [30], ensures several benefits. The registration of two images is domain-agnostic, which enables us to apply the medical algorithm to seismic data. The warp field is diffeomorphic, which ensures physically viable, topology-preserving warp velocity

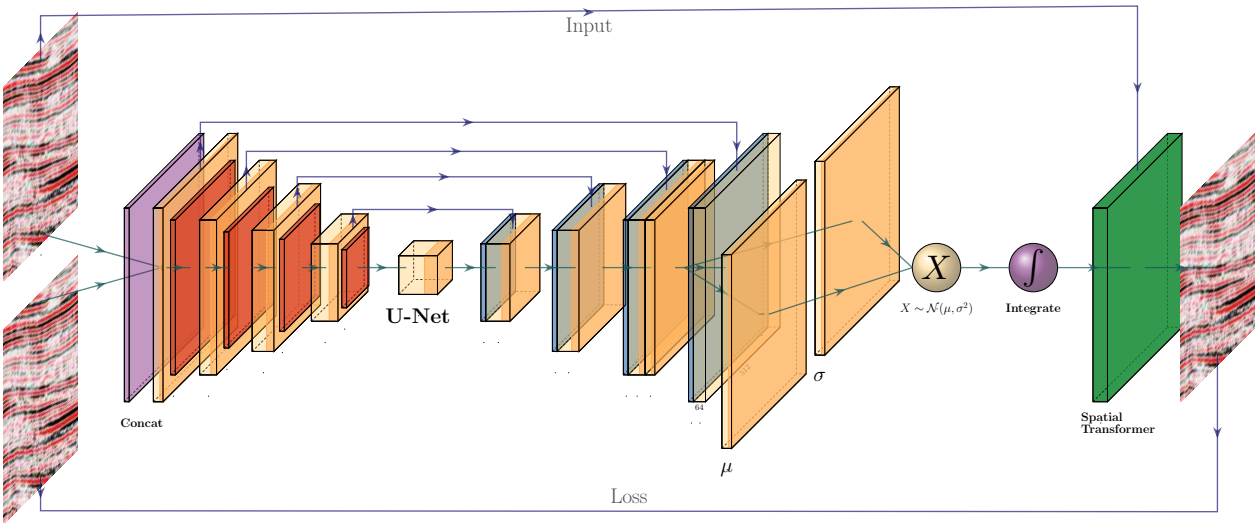


Fig. 2. 2D representation of Modified 3D Voxelmorph architecture to obtain full scale warp velocity field. The Encoder side of the U-Net architecture consists of four consecutive Convolutional (orange) and Pooling (red) layers, followed by a convolutional Bottleneck layer. The decoder of the U-Net architecture consists offour Upsampling (blue) and Convolutional layers are connected to the respective same size layers in the Encoder. The output is passed to two convolutional layers that are sampled by the reparametrization trick, to provide the static velocity field. The field is integrated via scaling and squaring and passed to the Spatial Transformer layer (green), which transforms the monitor to optimally match the base image, which is enforced by minimizing the mean squared error (MSE) of the images.

fields. Diffeomorphisms have recently gained great attention in the medical field, particularly with large deformation diffeomorphic metric mapping (LDDMM) [32], which is computationally expensive and has therefore not found great use in the wider field of geophysics, due to larger amounts of data. Moreover, this method implements a variational formulation based on the covariance of the flow field. 3D warping with uncertainty measure has not been used in seismic data before.

The network is implemented using Tensorflow [33] and Keras [34]. Our implementation is based on the original code in the Voxelmorph package [35].

IV. Experimental Results and Discussion

A. Experimental Setup

The experimental setup for this paper is based on a variation of the modified Voxelmorph [4] formulation. We extended the network to accept patches of data, because our seismic cubes are generally larger than the medical brain scans and therefore exceed the memory limits of our GPUs. Moreover, Voxelmorph in its original formulation provides sub-sampled flow fields, this is due to computational constraints. We decided to modify the network to provide full-scale flow fields, despite the computational cost. This enables direct interpretation of the warp field, which is common in 4D seismic analysis. However, we do provide an analysis in Section IV-B3 of the sub-sampled flow-field interpolated to full scale, in the way it would be passed to the Spatial Transformer layer.

The network definition for the subsampled flow field differs from the definition in Figure 2 that the last upsampling and convolution layer in the Unet, including the skip connection, right before the variational layers (μ, σ) is omitted. That

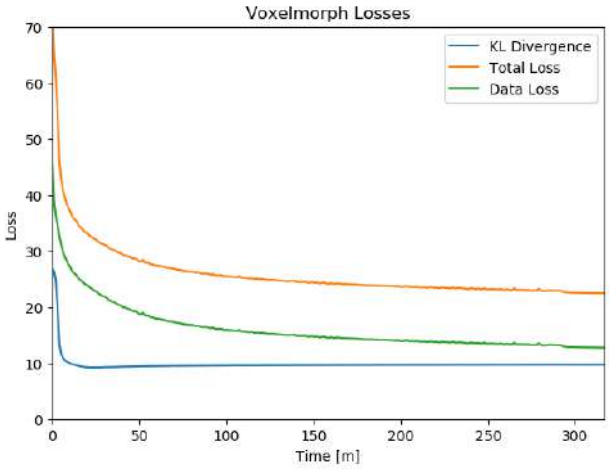


Fig. 3. Training Losses over time with the KL-divergence at the sampling layer, the data loss calculated by MSE, and the combined total loss.

leaves the flow field at a subsampled map by a factor of two. Computationally, this lowers the cost on the Integration operation before resampling for the Spatial Transformer.

The data situation for this experiment is special in the sense that the method is self-supervised. We therefore do not provide a validation dataset during training. The data are 6 surveys from the North Sea. Mail field from years 1088, 2005 A, 2005 B, and 2012. Further we compare to an adjecent field 1903 and 2005. While we would be content with the method working on the field data (years 1988 and 2005 Survey A) by itself, we do validate the results on separate data from

the same field which was acquired with different acquisition parameters and at different times (years 2005 Survey B and 2012). Moreover, we test the data on seismic data from an adjacent field that was acquired independently (years 1993 and 2005). All data is presented with a relative coordinate system due to confidentiality, where 0 s on the y-axis does not represent the actual onset of the recording. The field geology and therefore seismic responses are very different. Due to lack of availability we do not test the trained network on land data or data from different parts of the world. Considering, that the training set is one 4D seismic monitor-base pair, a more robust network would emerge from training on a variety of different cubes.

Figure 3 shows the training losses. Within a few epochs the network converges strongly, however within 10 epochs the KL divergence increases slightly over the training. The data loss, optimizing the warping result decreases over the training period. Private correspondence with the authors of Voxelmorph [35] suggests that a slight increase of the KL divergence is acceptable as long as the total loss decreases.

B. Results and Discussion

The network presented generates warp fields in three dimensions as well as uncertainty measures. We present results for three cases in Figure 4, 7, and 9 with the corresponding warp fieds and uncertainties in Figure 5, 8, and 10. In Figure 4 we show the results on the data, which the unsupervised method was trained on. Obtaining a warp field on the data itself is a good result, however, we additionally explore the generalizability of the method. Considering the network is trained to find an optimum warp field for the data it was originally trained on, we go on to test the network on data from the same field, that was recorded with significantly different acquisition parameters in Figure 7. These results test the networks generalizability on co-located data, therefore not expecting vastly differing seismic responses from the subsurface itself. The are imaging differences and differences in equipment in addition to the 4D difference however. In Figure 9 we use the network on unseen data from a different field. The geometry of the field, as well as the acquisition parameters are different, making generalization a challenge.

In Figure 4 we collect six 2D panels from the 3D warping operation. In Figure 4(a) and Figure 4(b) we show the unaltered base and monitor respectively. The difference between the unaltered cubes is shown in Figure 4(e). In Figure 4(c) we show the warped result by applying the z-warp field in Figure 4(d), as well as the warp fields in (x,y) direction fully displayed in Figure 5 including their respective uncertainties. The difference of the warped result in Figure 4(f) is calculated from the matched monitor in Figure 4(c) and the base in Figure 4(a).

It is apparent that the matched monitor significantly reduced noise by mis-aligned reflections. In Table I we present the numeric results. These were computed on the 3D cube for an accurate representation. We present the root mean square (RMS) and mean absolute error (MAE) and the according difference between Monitor and Matched Difference results.

Run	Monitor RMS	Matched RMS	Ratio %	Monitor MAE	Matched MAE	Ratio %
Train	0.1047	0.0525	50.1	0.0744	0.0348	46.7
Test A	0.0381	0.0237	62.2	0.0291	0.0172	59.1
Test B	0.0583	0.0361	62.0	0.0451	0.0254	56.4

TABLE I
QUANTITATIVE EVALUATION OF RESULTS. RMS AND MAE CALCULATED AGAINST RESPECTIVE BASE DATA. TRAINING RECALL, TEST A - SAME FIELD, DIFFERENT ACQUISITION, TEST B - DIFFERENT FIELD, DIFFERENT ACQUISITION

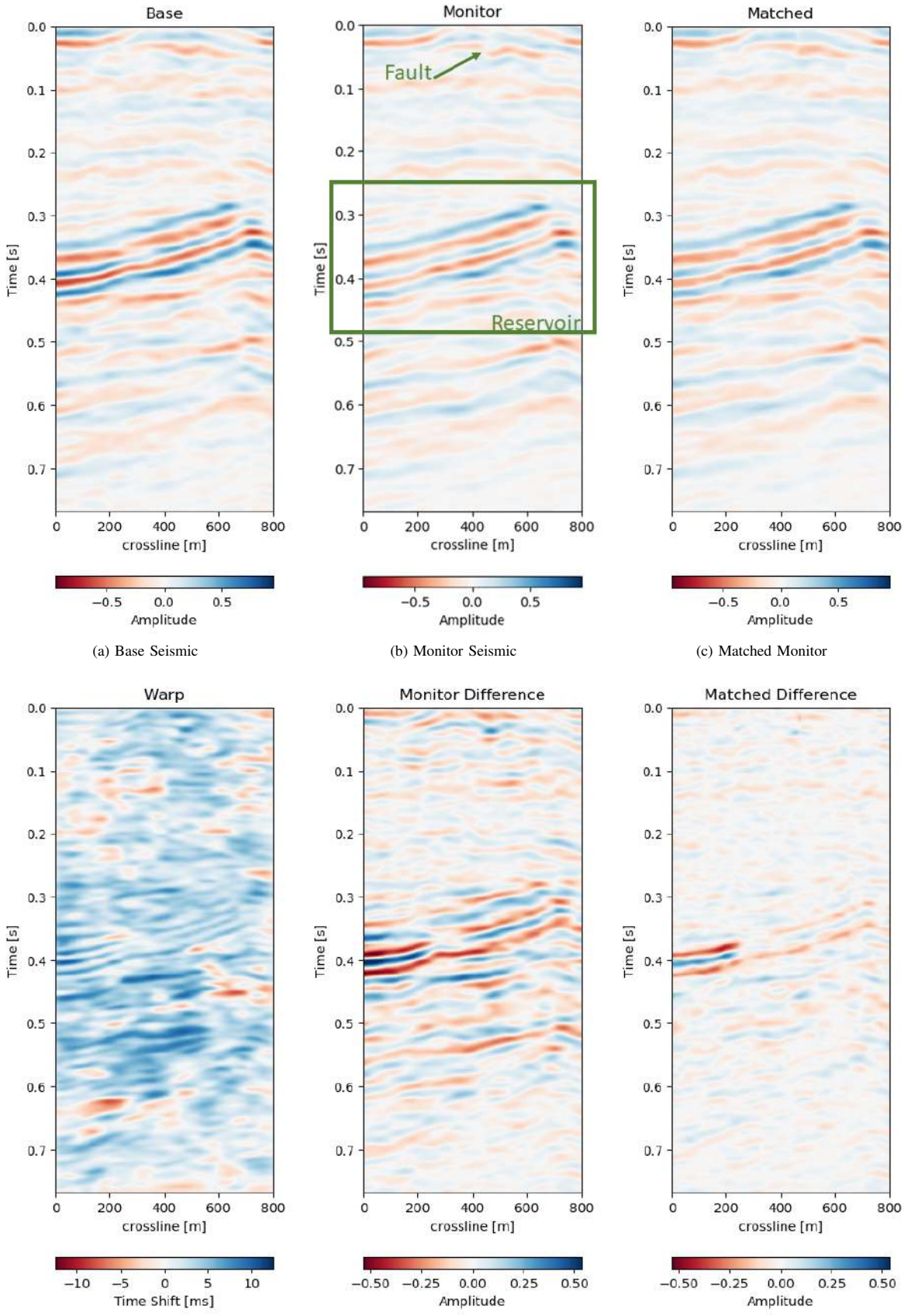
We present RMS and MAE to make the values comparable in magnitude as opposed the mean squared error (MSE). We present both values, because the RMS value is more sensitive to large values, while MAE scales the error linearly therefore not masking low amplitude mis-alignments. Both measurements show a reduction on the train data to 50% or below. The test on both the validation data on the same field and the test data on another field show a similar reduction, while the absolute error differs in a stable manner.

In Figure 5 we present the three dimensional warp field to accompany the results in Figure 4. Figure 5(a), 5(b), and 5(c) show the warp field in x, y, and z-direction. The z-direction is generally referred to as time shifts in 4D seismic. Figure 5(d), 5(e), and 5(f) contain the corresponding uncertainties in x, y, and z-direction obtained from the network.

1) Recall to Training Data: In Figure 4 we evaluate the results of the self-supervised method on the training data itself. The main focus is on the main reflector in the center of the panels. The difference in Figure 4(e) shows that the packet of reflectors marked reservoir in the monitor is out of alignment, causing a large difference, which is corrected for in Figure 4(f). The topmost section in the panel of Figure 4(c) shows the alignment of a faulted segment, marked fault in the monitor, to an unfaulted segment in the base. The fault appearing is most likely due to vastly improved acquisition technology for the monitor.

The warp fields in Figure 5 are an integral part in QC-ing the validity of the results. Physically, we expect the strongest changes in the z-direction in Figure 5(c). The changes in Figure 5(a) and Figure 5(b) show mostly sub-sampling magnitude shifts, except for the x-direction shifts around the fault in the top-most panel present in the monitor in Figure 4(b). Figure 5(a) and Figure 5(b) show strong shifts at 0.4s on the left of the panel which corresponds to the strong amplitude changes in the base and monitor. On the one side these correspond to the strongest difference section, additionally these are geological hinges, which are under large geomechanical strain. However, these are very close to the sides of the warp, which may cause artifacts. Figure 5(d), Figure 5(e), and Figure 5(f) show the uncertainty of the network. These uncertainties are across the bank within the 10% range of the sampling rate ($\Delta t = 4$ ms, $\Delta x, y = 25$ m). The certainty within the bulk package in the center of the panels is the lowest in x-, y-, and z-direction. While being relatively lover in the problematic regions discussed before.

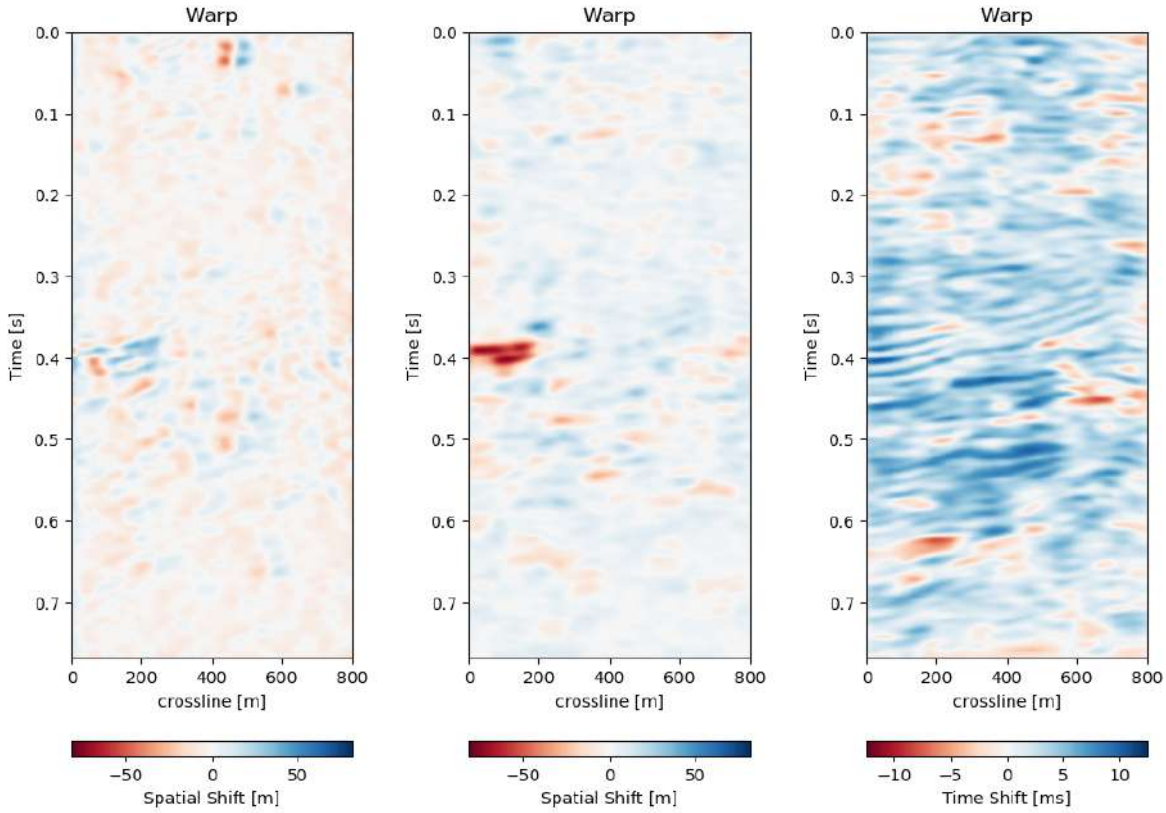
The warp field in Figure 5(d) contains some reflector shaped warp vectors around 0.4 s, which is due to the wavelet mismatch of the 1988 base to the 2005 monitor. The diffeomorphic



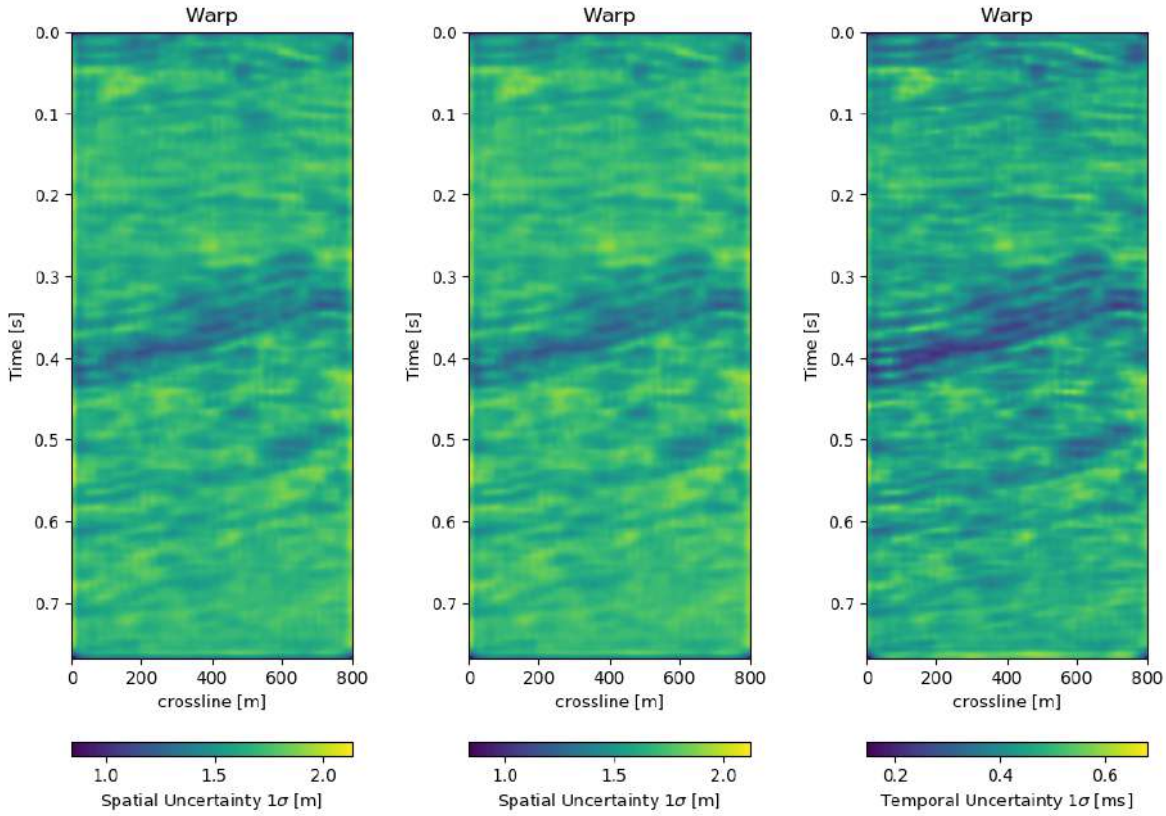
(a) Base Seismic (b) Monitor Seismic (c) Matched Monitor

(d) z-Direction Shifts (e) Difference Monitor-Base (f) Difference Matched-Base

Fig. 4. Warp results and change in difference on training recall of 1988 to 2005a data. Axes are relative to comply with confidentiality.



(a) x-direction Shifts (b) y-direction Shifts (c) z-direction Shifts



(d) x-direction Uncertainty (e) y-direction Uncertainty (f) z-direction Uncertainty

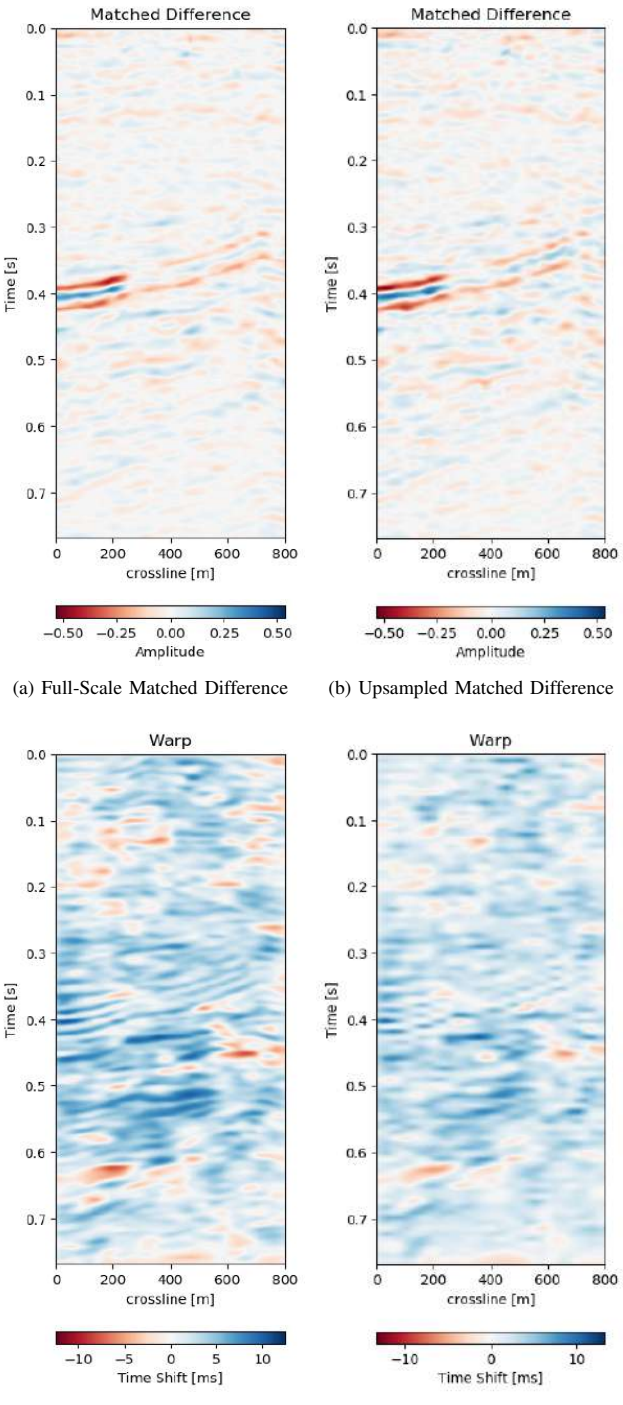
Fig. 5. Warp fields (top) with uncertainties (bottom) that accompanies training recall in Figure 4

nature of the network aligns the reflectors in the image, which causes some reflector artifacts in the z-direction maps.

2) Generalization of the Network: While the performance of the method on a data set by itself is good, obtaining a trained model that can be applied on other similar data sets is essential even for self-supervised methods. We test the network on two test sets, Test A is conducted on the same geology with unseen data from a different acquisition, while Test B is on a different field and a different acquisition. The network was trained on a single acquisition relation (2005a - 1988). In Figure 7 we present the crossline data from the same field the network was trained on. The data sets was however acquired at a different calendar times (2005b - 2012), with different acquisition parameters. It follows that although the geology and therefore the reflection geometry is similar, the wavelet and hence the seismic response are vastly different. This becomes apparent when comparing the base Figure 7a to Figure 4(b), which were acquired in the same year.

Test A evaluates the network performance on unseen data in the same field (Train: 1988-2005a, Test A: 2005b - 2012). The quantitative results in Table I for Test A generally show lower absolute errors compared to the training results in Section IV-B1. The reduction of the overall amplitudes in the difference maps is reduce by 40%. The unaligned monitor difference in Figure 7(e) shows a strong coherent difference around below the main packet of reflectors around 0.3 s to 0.4 s. This would suggest a velocity draw-down in this packet. While the top half of the unaligned difference contains some misalignment, we would expect the warp field to display a shift around 0.35 s, which can be observed in Figure 7(d). The aligned difference in Figure 7(f) contains less coherent differences. The difference does still show some overall noise in the maps. This could be improved upon by a more diverse training set. The higher resolution data from 2005 and 2012 possibly has an influence on the result too. Regardless, we can see some persisting amplitude difference around 0.4 s which appears to be signal as opposed to some misalignment noise above. The warp fields in Figure 8 show relatively smooth warp fields in x- and y-direction. The warp field in Figure 8(f) shows overall good coherence, including the change around 0.4 s we would expect. The uncertainty values are in sub-sampling range, with the strongest certainty within the strong reflector packet at 0.35 s.

Test B evaluates the network performance on a different field at different times. The test shows a very similar reduction of overall errors in Table I. The RMS is reduced by 38% and the MAE is reduced more slightly more in comparison to Test A. In Figure 9 we present the seismic panels to accompany Test B. The data in Figure 9(a) and Figure 9(b) is well resolved and shows good coherence. However, the unaligned difference in Figure 9(e) shows very strong variations in the difference maps. Figure 9(f) reduces these errors significantly, bringing out coherent differences in the main reflector at 0.27 s. We can see strong chaotic differences in Figure 9(e), due to the faulted nature of the geology. The network aligns these faulted blocks relatively well, however, some artifacts persist. This is consistent with the warp fields in Figure 10. The x- and y-direction in Figure 10(d) and Figure 10(e) respectively



(a) Full-Scale Matched Difference (b) Upsampled Matched Difference

(c) Full-Scale Warp Field (d) Upsampled Warp Field

Fig. 6. Comparison of matched differences (top) and z-direction warp field (bottom) of full-scale neural architecture (left) and subsampled neural architecture (right).

show overall smooth changes, around faults, these changes are stronger. The z-direction changes are consistent with the Training validation and Test A, where the changes are overall stronger. This is also consistent with our geological intuition.

3) Subsampled Flow: The original Voxelmorph implementation uses a subsampled warp field. This has two benefits,

namely a smoother warp velocity field and reduced computational cost. The aforementioned results were obtained using a full-scale network. In Figure 6 we present the full scale and upsampled results on the training set. The matched difference in Figure 6b contains more overall noise compared to Figure 6a. This is congruent with the warp fields in the figure. The upsampled z-direction warp field in Figure 6d seems to have some aliasing on the diagonal reflector around 0.4 s. This explains some of the artifacts in the difference in Figure 6b. The overall warp velocity in Figure 6d is smoother compared to the full-scale field. However, the general structure of coherent negative and positive areas matches in both warp fields, while the details differ. The main persistent difference of the reflector packet at 0.4 s seems similar, nevertheless, the differences further up slope to the right are smoother in the full scale network result and have stronger residual amplitudes in the upsampled network.

V. Conclusion

We introduce a deep learning based self-supervised 4D seismic warping method. Currently, time shifts are most commonly estimated in 1D due to computational constraints. We explore 3D time-shift estimation as a viable alternative, which decouples imaging and acquisition effects, geomechanical movement and changes in physical properties like velocity and porosity from confounding into a single dimension. Existing 3D methods are computationally expensive, where this learnt model can generalize to unseen data without re-training, with calculation times within minutes on consumer hardware. Moreover, this method supplies invertible, reproducible, dense 3D alignment while providing warp fields with uncertainty measures, while leveraging recent advancements in neural networks and deep learning.

We evaluate our network on the training data and two different independent test sets. We do not expect the aligned difference to be exactly zero, due to actual physical changes in the imaged subsurface. Although the network is unsupervised, a transfer to unseen data is desirable and despite some increase in the overall error possible. The warping on the training data is very good and the warp fields are coherent and reflect the physical reality one would expect. The transfer too unseen data works well, although the misalignment error increases. The decrease in both RMS and MAE is consistent across test sets.

Furthermore, we implement a variational scheme which provides uncertainty measures for the time shifts. On the data presented, we obtain subsample scale uncertainties across all directions. The main assumption of the network is a diffeomorphic deformation, which is topology preserving. We show that the network handles faults well in both training recall and test data, that in theory could violate the diffeomorphic assumption.

We go on to compare a full-scale network to an upsampled network. The full-scale network yields better results and is preferable on seismic data in comparison to the upsampled network presented in the original medical Voxelmorph.

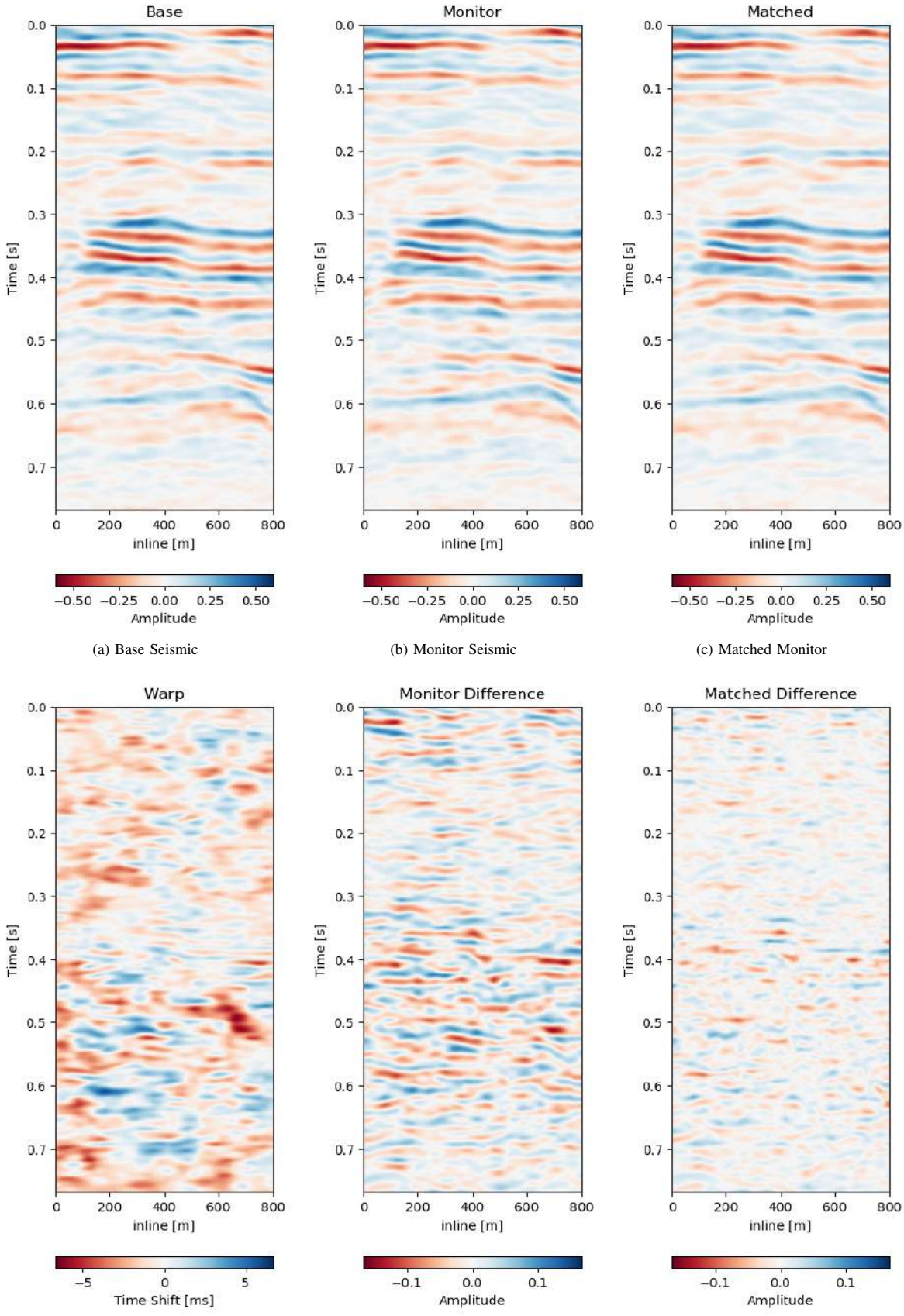
We do expect the network to improve upon training on a more diverse variety of data sets and seismic responses. While the initial training is time-consuming (25 h on a Nvidia Titan X with Pascal chipset), inference is near instantaneous. Moreover, transfer of the trained network to a new data set is possible without training, while accepting some error. Alternatively fine-tuning to new data is possible within few epochs ($<$1 h).

Acknowledgment

The research leading to these results has received funding from the Danish Hydrocarbon Research and Technology Centre under the Advanced Water Flooding program. We thank DTU Compute for access to the GPU Cluster. We thank Total E&P Denmark for permission to use the data and publish examples.

References

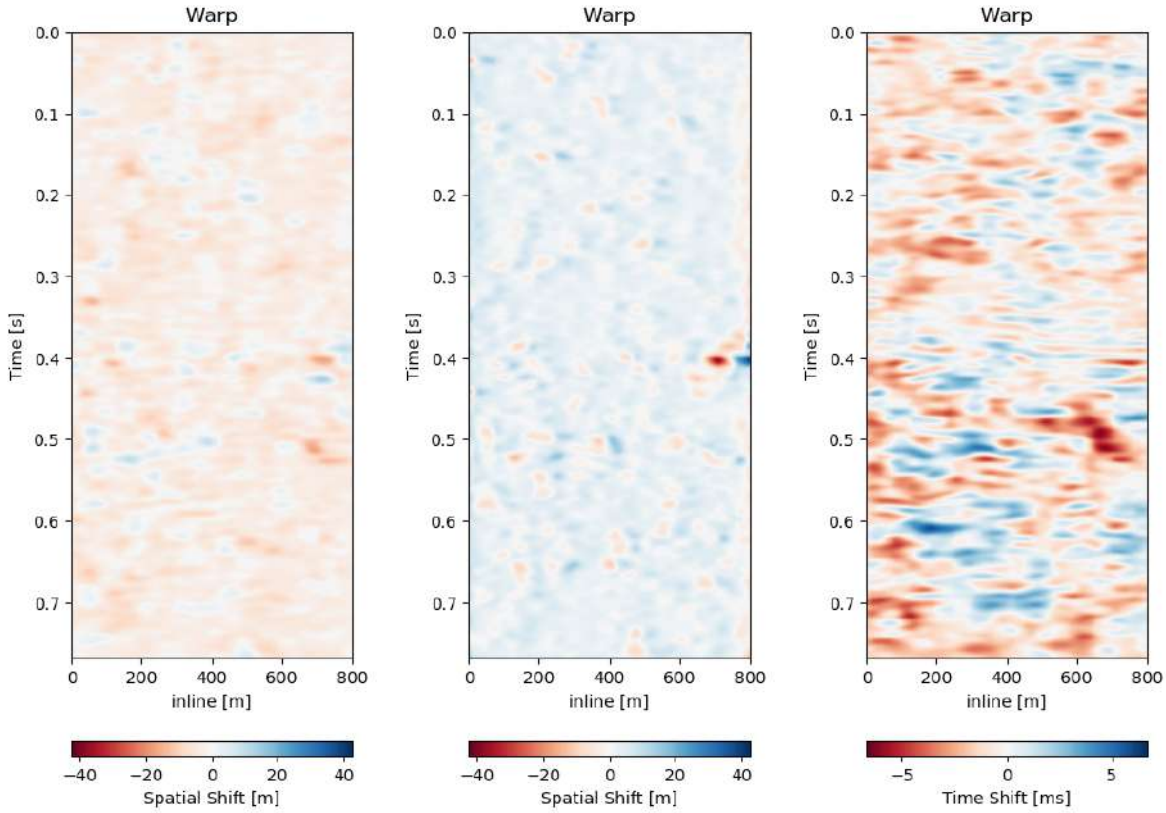
[1] C. MacBeth, M.-D. Mangriotis, and H. Amini, "Post-stack 4d seismic time-shifts: Interpretation and evaluation," *Geophysical Prospecting*, vol. 67, no. 1, pp. 3–31, 2019.
[2] A. Wirgin, "The inverse crime," *arXiv preprint math-ph/0401050*, 2004.
[3] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Advances in Neural Information Processing Systems*, 2015, pp. 2575–2583.
[4] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "Voxelmorph: a learning framework for deformable medical image registration," *IEEE transactions on medical imaging*, 2019.
[5] P. Hatchell and S. Bourne, "Rocks under strain: Strain-induced time-lapse time shifts are observed for depleting reservoirs," *The Leading Edge*, vol. 24, no. 12, pp. 1222–1225, 2005.
[6] D. Hale, "Dynamic warping of seismic images," *Geophysics*, vol. 78, no. 2, pp. S105–S115, 2013.
[7] E. Zabihi Naeini, H. Hoeber, G. Poole, and H. R. Siahkoohi, "Simultaneous multivintage time-shift estimation," *Geophysics*, vol. 74, no. 5, pp. V109–V121, 2009.
[8] S. A. Hall, C. MacBeth, O. I. Barkved, and P. Wild, "Cross-matching with interpreted warping of 3d streamer and 3d ocean-bottom-cable data at valhall for time-lapse assessment," *Geophysical Prospecting*, vol. 53, no. 2, pp. 283–297, 2005.
[9] D. Hale, "An efficient method for computing local cross-correlations of multi-dimensional signals," *CWP Report*, vol. 656, 2006.
[10] J. Rickett, L. Duranti, T. Hudson, B. Regel, and N. Hodgson, "4d time strain and the seismic signature of geomechanical compaction at genesis," *The Leading Edge*, vol. 26, no. 5, pp. 644–647, 2007.
[11] A. Cherrett, I. Escobar, and H. Hansen, "Fast deterministic geostatistical inversion," in *73rd EAGE Conference and Exhibition incorporating SPE EUROPEC 2011*, 2011.
[12] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.
[13] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4161–4170.
[14] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.
[15] T.-W. Hui, X. Tang, and C. Change Loy, "Liteflownet: A lightweight convolutional neural network for optical flow estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8981–8989.
[16] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8934–8943.
[17] D. Gadot and L. Wolf, "Patchbatch: A batch augmented loss for optical flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4236–4245.
[18] F. Güney and A. Geiger, "Deep discrete flow," in *Asian Conference on Computer Vision*. Springer, 2016, pp. 207–224.



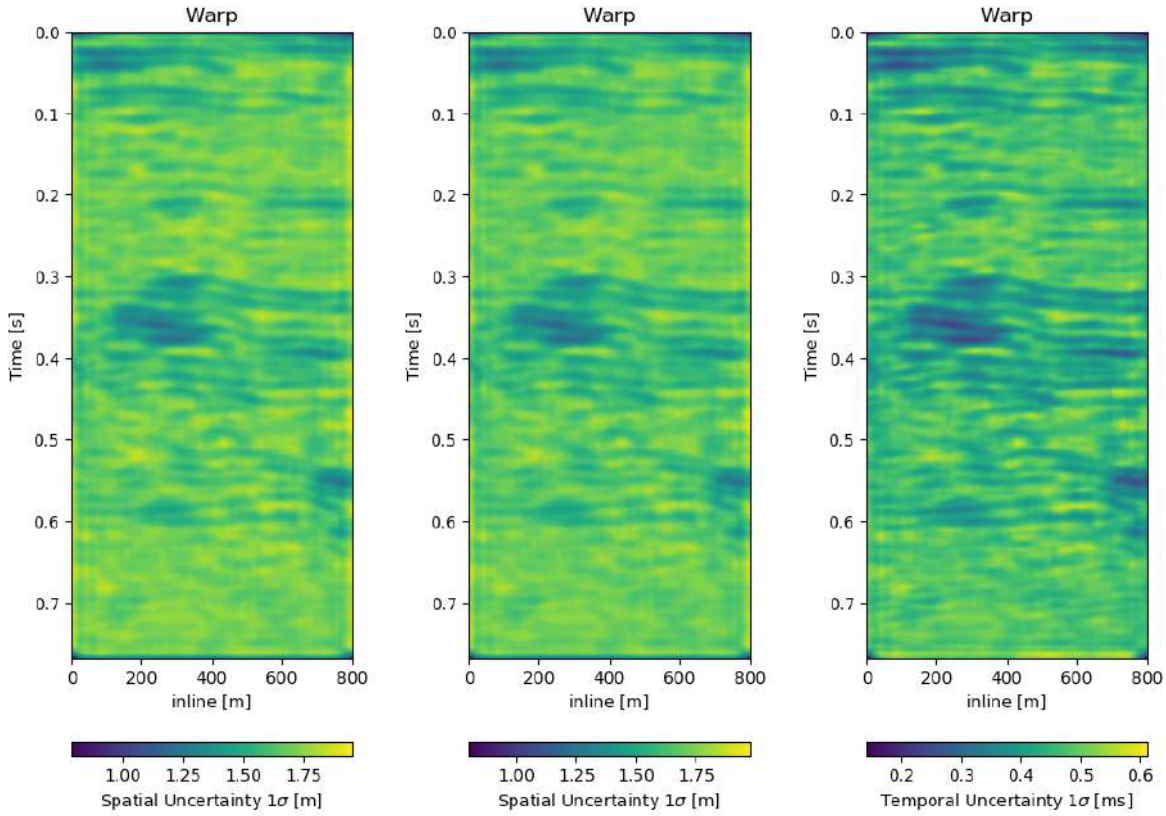
(a) Base Seismic (b) Monitor Seismic (c) Matched Monitor

(d) z-Direction Shifts (e) Difference Monitor-Base (f) Difference Matched-Base

Fig. 7. Matched difference and warp field for generalization of network to same field with different data (2005b and 2012).

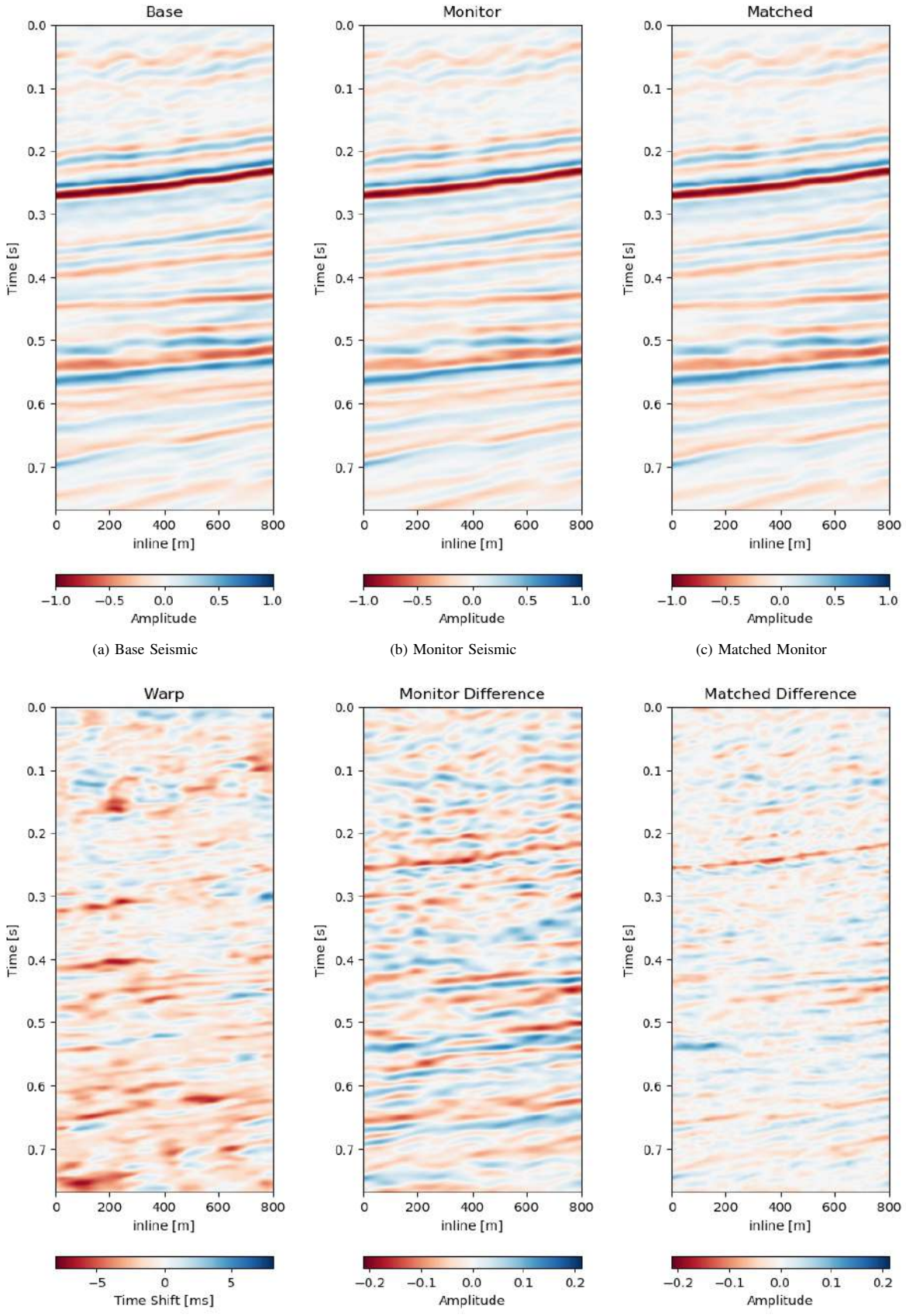


(a) x-direction Shifts (b) y-direction Shifts (c) z-direction Shifts



(d) x-direction Uncertainty (e) y-direction Uncertainty (f) z-direction Uncertainty

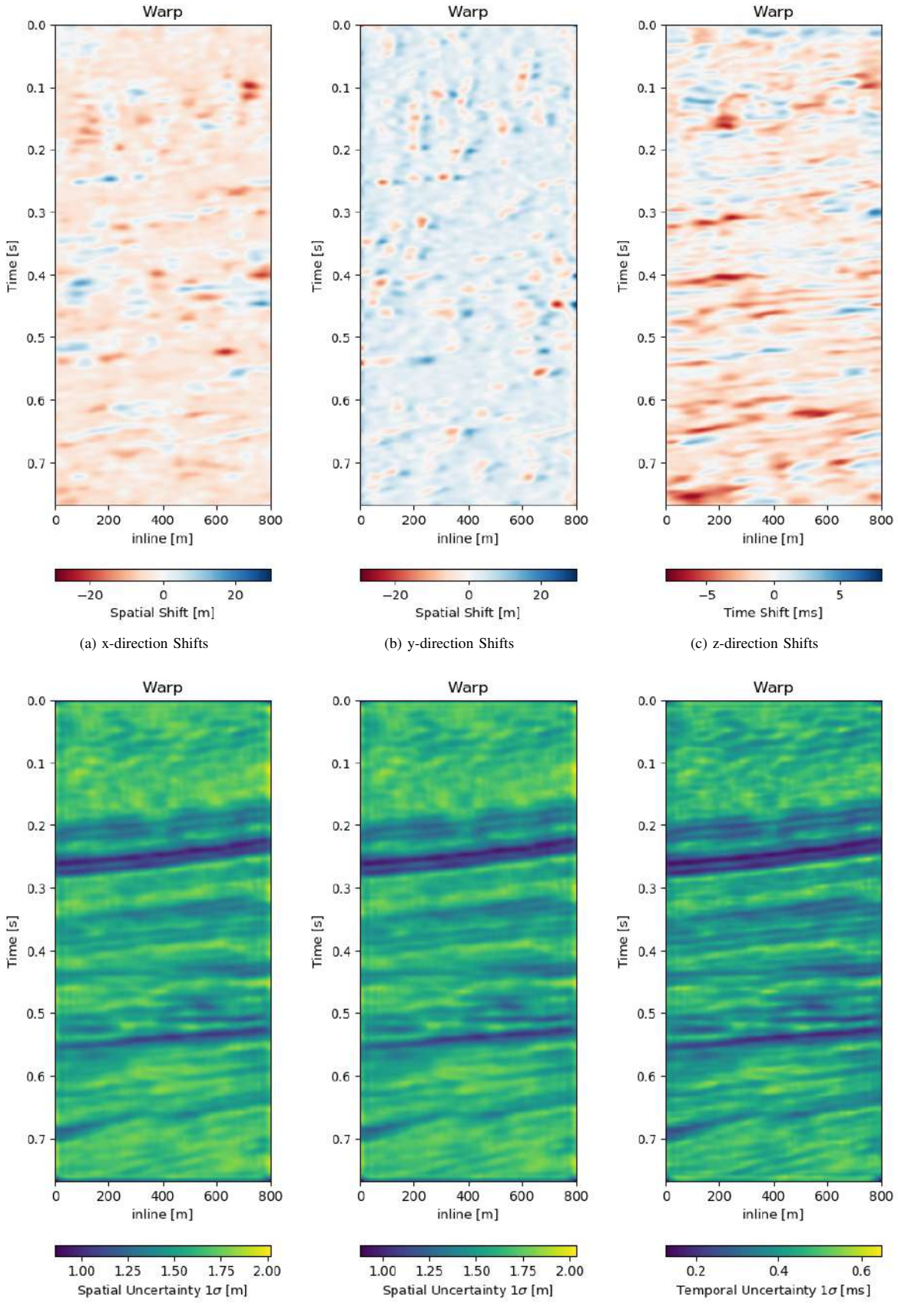
Fig. 8. Warp fields (top) with uncertainties (bottom) that accompanies same field generalization in Figure 7



(a) Base Seismic (b) Monitor Seismic (c) Matched Monitor

(d) z-Direction Shifts (e) Difference Monitor-Base (f) Difference Matched-Base

Fig. 9. Matched difference and warp field for generalization of network to a different field (1993 and 2005).



(a) x-direction Shifts (b) y-direction Shifts (c) z-direction Shifts

(d) x-direction Uncertainty (e) y-direction Uncertainty (f) z-direction Uncertainty

Fig. 10. Warp fields (top) with uncertainties (bottom) that accompanies generalization to different field in Figure 9

APPENDIX A

ImageNet Results

Name	Citation	Top-1 [%]	Top-5 [%]	Param [M]
AlexNet	Krizhevsky et al. (2012)	63,3	84,6	60
VGG-16	Simonyan et al. (2014)	74,4	91,9	138
VGG-19	Simonyan et al. (2014)	74,5	92,0	144
AmoebaNet-B	Real et al. (2019)	82,3	96,1	84
AmoebaNet-C	Real et al. (2019)	83,1	96,3	155,3
DenseNet-201	Huang et al. (2017a)	78,5	94,4	20
EfficientNet-B1	Tan et al. (2019b)	78,8	94,4	7,8
EfficientNet-B2	Tan et al. (2019b)	79,8	94,9	9,2
EfficientNet-B3	Tan et al. (2019b)	81,1	95,5	12
EfficientNet-B4	Tan et al. (2019b)	82,6	96,3	19
EfficientNet-B5	Tan et al. (2019b)	83,3	96,7	30
EfficientNet-B6	Tan et al. (2019b)	84,0	96,9	43
EfficientNet-B7	Tan et al. (2019b)	85,0	97,2	66
Inception V1	Szegedy et al. (2015)	69,8	89,9	5
Inception V2	Ioffe et al. (2015)	74,8	92,2	11,2
Inception V3	Szegedy et al. (2016)	78,8	94,4	23,8
InceptResNet V2	Szegedy et al. (2017)	80,1	95,1	55,8
MixNet-S	Tan et al. (2019c)	75,8	92,8	4,1
MixNet-M	Tan et al. (2019c)	77,0	93,3	5
MixNet-L	Tan et al. (2019c)	78,9	94,2	7,3
NasNet-A6	Zoph et al. (2018)	82,7	96,2	89
MNasNet-A1	Tan et al. (2019a)	76,7	93,3	5,2
MNasNet-A3	Tan et al. (2019a)	75,2	92,5	3,9
FixPNasNet-5	Touvron et al. (2019)	83,7	96,8	86,1
ResNet-50-D	He et al. (2019)	77,1	93,5	25
FixResNet-50	Touvron et al. (2019)	79,1	94,6	25,6
ResNet-101	He et al. (2016)	78,2	93,9	40
ResNeXt-101	Xie et al. (2017)	80,9	95,6	83,6
Oct-ResNet-152	Chen et al. (2019)	82,9	96,3	67

Table A.1: ImageNet results of different neural network architectures (Partial resource from Papers With Code).

APPENDIX B

Publications of Neural Networks in Geoscience

Topic	Publications
Digital Rock Model	Mosser et al. (2017), Mosser et al. (2018b), Sudakov et al. (2018), and Mosser et al. (2018a)
First Break Picking	Murat et al. (1992), McCormack et al. (1993), Dai et al. (1995), Dai et al. (1997a), and Ross et al. (2018a)
Ground Penetrating Radar	Al-Nuaimy et al. (2000), Gamba et al. (2000), Shihab et al. (2002a), Shihab et al. (2002b), Youn et al. (2002), Birkenfeld (2010), Cui et al. (2010), Maas et al. (2013), Núñez-Nieto et al. (2014), Mertens et al. (2016), Hansen et al. (2017), and Kilic et al. (2018)
Mineral Prospectivity Mapping	Porwal et al. (2003), Oh et al. (2010), Chen et al. (2014), Chen (2015), and Jafrasteh et al. (2016)
Property Estimation	Bagheripour (2014), Iturrarán-Viveros et al. (2014), Boateng et al. (2017), and Kuroda et al. (2016)
Seismic Deconvolution	Zhao et al. (1988), Wang et al. (1997), Calderón-Macías et al. (1997), and Harrigan et al. (1991)
Seismic Horizon Picking	Huang et al. (1990), Legget et al. (1996), Zhang et al. (2001), and Leggett et al. (2003)

Automatic Seismic Interpretation	Meldahl et al. (2001), Strecker et al. (2002), Klose (2006), Zheng et al. (2014), Marroquín (2014), Qi et al. (2016), Zhao et al. (2016), Roden et al. (2015), Huang et al. (2017b), Lewis et al. (2017), Waldeland et al. (2017), Guo et al. (2017), Zhao et al. (2017a), Veillard et al. (2018), Araya-Polo et al. (2017), Dramsch et al. (2018c), Chevitarese et al. (2018), Gramstad et al. (2018), Guitton (2018), Purves et al. (2018), Shafiq et al. (2018a), Shafiq et al. (2018b), Waldeland et al. (2018), AlRegib et al. (2018), Le Bouteiller et al. (2018), Li et al. (2018), Sacrey et al. (2018), Shafiq et al. (2018c), and Wu et al. (2018)
Seismic Inversion	Röth et al. (1994), Langer et al. (1996), Iturrarán-Viveros (2012), Ansari (2014), Verma et al. (2014), Golsanami et al. (2015), Schuster (2018), Araya-Polo et al. (2018), Mosser et al. (2018d), Mosser et al. (2018c), and Richardson (2018)
Seismic Processing	McCormack et al. (1993), Ashida (1996), Patel et al. (2016), and Bhowmick et al. (2018)
Seismic Tomography	Bauer et al. (2008) and Braeuer et al. (2015)
Seismic Well-Tie	Chaki et al. (2018)
Seismology	Dowla et al. (1990), Romeo (1994), Musil et al. (1996a), Musil et al. (1996b), Falsaperla et al. (1996), Dai et al. (1997b), Wang et al. (1997), Langer et al. (1996), Scarpetta et al. (2005), Esposito et al. (2006), Gentili et al. (2006), Langer et al. (2003), Meier et al. (2007b), Meier et al. (2007a), Castellaro et al. (2007), Draelos et al. (2015), Ross et al. (2018b), and Zhu et al. (2018)
Well-Log analysis	Huang et al. (1996), Fung et al. (1997), Bhatt et al. (2002), Helle et al. (2002), Asoodeh et al. (2014), Anifowose et al. (2017b), Saporetti et al. (2018), Maiti et al. (2010), Chang et al. (2002), Bauer et al. (2015), Emelyanova et al. (2017), and Carreira et al. (2018)
Vertical Seismic Profiling	Dai et al. (1994)

Table B.1: Neural Networks in Geoscience.

APPENDIX C

Software Manual: Keras Complex

Software manual from the `keras complex` package for complex-valued neural networks in Python 3. Original code by Trabelsi et al. (2017) in Theano. Code ported to Tensorflow, consolidated, packaged, set up with automatic testing and documentation by Dramsch et al. (2019c).

Original Code:

C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal (2017). "Deep complex networks". In: *arXiv preprint arXiv:1705.09792*

Port to Keras with Tensorflow:

J. S. Dramsch and Contributors (2019c). *Complex-Valued Neural Networks in Keras with Tensorflow*. Open-Source Software. DOI: 10.6084/m9.figshare.9783773. URL: https://github.com/JesperDramsch/keras-complex

Keras Complex

Oct 16, 2019

Table of Contents

Complex-valued convolutions could provide some interesting results in signal processing-based deep learning. A simple(-ish) idea is including explicit phase information of time series in neural networks. This code enables complex-valued convolution in convolutional neural networks in keras with the TensorFlow backend. This makes the network modular and interoperable with standard keras layers and operations.

Keras Complex

CHAPTER 1

Contents

1.1 Introduction

Complex-valued convolutions could provide some interesting results in signal processing-based deep learning. A simple(-ish) idea is including explicit phase information of time series in neural networks. This code enables complex-valued convolution in convolutional neural networks in keras with the TensorFlow backend. This makes the network modular and interoperable with standard keras layers and operations.

1.2 Installation

Installation is as easy as

```
pip install keras-complex
```

but you'll need to install tensorflow in addition using

```
pip install tensorflow-gpu
```

for the GPU version or for the non-GPU version:

```
pip install tensorflow
```

1.3 complexnn

1.3.1 complexnn package

Submodules

complexnn.bn module

complexnn.bn.**ComplexBN**(*input_centred*, *Vrr*, *Vii*, *Vri*, *beta*, *gamma_rr*, *gamma_ri*, *gamma_ii*, *scale=True*, *center=True*, *layernorm=False*, *axis=-1*)

Complex Batch Normalization

Arguments: input_centred – input data Vrr – Real component of covariance matrix V Vii – Imaginary component of covariance matrix V Vri – Non-diagonal component of covariance matrix V beta – Lernable shift parameter beta gamma_rr – Scaling parameter gamma - rr component of 2x2 matrix gamma_ri – Scaling parameter gamma - ri component of 2x2 matrix gamma_ii – Scaling parameter gamma - ii component of 2x2 matrix

Keyword Arguments: scale {bool} {bool} – Standardization of input (default: {True}) center {bool} – Mean-shift correction (default: {True}) layernorm {bool} – Normalization (default: {False}) axis {int} – Axis for Standardization (default: {-1})

Raises: ValueError: Dimonsional mismatch

Returns: Batch-Normalized Input

class complexnn.bn.**ComplexBatchNormalization**(*axis=-1*, *momentum=0.9*, *epsilon=0.0001*, *center=True*, *scale=True*, *beta_initializer='zeros'*, *gamma_diag_initializer='sqrt_init'*, *gamma_off_initializer='zeros'*, *moving_mean_initializer='zeros'*, *moving_variance_initializer='sqrt_init'*, *moving_covariance_initializer='zeros'*, *beta_regularizer=None*, *gamma_diag_regularizer=None*, *gamma_off_regularizer=None*, *beta_constraint=None*, *gamma_diag_constraint=None*, *gamma_off_constraint=None*, ***kwargs*)

Bases: keras.engine.base_layer.Layer

Complex version of the real domain Batch normalization layer (Ioffe and Szegedy, 2014). Normalize the activations of the previous complex layer at each batch, i.e. applies a transformation that maintains the mean of a complex unit close to the null vector, the 2 by 2 covariance matrix of a complex unit close to identity and the 2 by 2 relation matrix, also called pseudo-covariance, close to the null matrix. # Arguments

axis: Integer, the axis that should be normalized (typically the features axis). For instance, after a *Conv2D* layer with *data_format="channels_first"*, set *axis=2* in *ComplexBatchNormalization*.

momentum: Momentum for the moving statistics related to the real and imaginary parts.

epsilon: Small float added to each of the variances related to the real and imaginary parts in order to avoid dividing by zero.

center: If True, add offset of *beta* to complex normalized tensor. If False, *beta* is ignored. (beta is formed by real_beta and imag_beta)

scale: If True, multiply by the *gamma* matrix. If False, *gamma* is not used.

beta_initializer: Initializer for the real_beta and the imag_beta weight. gamma_diag_initializer: Initializer for the diagonal elements of the gamma matrix.

which are the variances of the real part and the imaginary part.

gamma_off_initializer: Initializer for the off-diagonal elements of the gamma matrix. moving_mean_initializer: Initializer for the moving means. moving_variance_initializer: Initializer for the moving variances. moving_covariance_initializer: Initializer for the moving covariance of

the real and imaginary parts.

beta_regularizer: Optional regularizer for the beta weights. gamma_regularizer: Optional regularizer for the gamma weights. beta_constraint: Optional constraint for the beta weights. gamma_constraint: Optional constraint for the gamma weights.

 # Input shape Arbitrary. Use the keyword argument *input_shape* (tuple of integers, does not include the samples axis) when using this layer as the first layer in a model.

 # Output shape Same shape as input.

 # References

- [Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift](https://arxiv.org/abs/1502.03167)

build(*input_shape*)
Creates the layer weights.

Must be implemented on all layers that have weights.

 # Arguments

input_shape: Keras tensor (future input to layer) or list/tuple of Keras tensors to reference for weight shape computations.

call(*inputs*, *training=None*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

get_config()
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

complexnn.bn.**complex_standardization**(*input_centred*, *Vrr*, *Vii*, *Vri*, *layernorm=False*, *axis=-1*)
Complex Standardization of input

Arguments: input_centred – Input Tensor Vrr – Real component of covariance matrix V Vii – Imaginary component of covariance matrix V Vri – Non-diagonal component of covariance matrix V

Keyword Arguments: layernorm {bool} – Normalization (default: {False}) axis {int} – Axis for Standardization (default: {-1})

Raises: ValueError: Mismatched dimensoins

Returns: Complex standardized input

complexnn.bn.**sanitizedInitGet**(*init*)

complexnn.bn.**sanitizedInitSer**(*init*)

complexnn.bn.**sqrt_init**(*shape*, *dtype=None*)

complexnn.conv module

conv.py

class complexnn.conv.**ComplexConv**(*rank*, *filters*, *kernel_size*, *strides=1*, *padding='valid'*, *data_format=None*, *dilation_rate=1*, *activation=None*, *use_bias=True*, *normalize_weight=False*, *kernel_initializer='complex'*, *bias_initializer='zeros'*, *gamma_diag_initializer=<function sqrt_init>*, *gamma_off_initializer='zeros'*, *kernel_regularizer=None*, *bias_regularizer=None*, *gamma_diag_regularizer=None*, *gamma_off_regularizer=None*, *activity_regularizer=None*, *kernel_constraint=None*, *bias_constraint=None*, *gamma_diag_constraint=None*, *gamma_off_constraint=None*, *init_criterion='he'*, *seed=None*, *spectral_parametrization=False*, *transposed=False*, *epsilon=1e-07*, ***kwargs*)
Bases: keras.engine.base_layer.Layer

Abstract nD complex convolution layer.

This layer creates a complex convolution kernel that is convolved with the layer input to produce a tensor of outputs. If *use_bias* is True, a bias vector is created and added to the outputs. Finally, if *activation* is not *None*, it is applied to the outputs as well.

Arguments:

rank: Integer, the rank of the convolution, e.g., "2" for 2D convolution.

filters: Integer, the dimensionality of the output space, i.e., the number of complex feature maps. It is also the effective number of feature maps for each of the real and imaginary parts. (I.e., the number of complex filters in the convolution) The total effective number of filters is 2 x filters.

kernel_size: An integer or tuple/list of n integers, specifying the dimensions of the convolution window.

strides: An integer or tuple/list of n integers, specifying the strides of the convolution. Specifying any stride value != 1 is incompatible with specifying any *dilation_rate* value != 1.

padding: One of *"valid"* or *"same"* (case-insensitive). data_format: A string, one of *channels_last* (default) or

channels_first. The ordering of the dimensions in the inputs. *channels_last* corresponds to inputs with shape *(batch, ..., channels)* while *channels_first* corresponds to inputs with shape *(batch, channels, ...)*. It defaults to the *image_data_format* value found in your Keras config file at *~/.keras/keras.json*. If you never set it, then it will be "channels_last".

dilation_rate: An integer or tuple/list of n integers, specifying the dilation rate to use for dilated convolution. Currently, specifying any *dilation_rate* value != 1 is incompatible with specifying any *strides* value != 1.

activation: Activation function to use (see keras.activations). If you don't specify anything, no activation is applied (i.e., "linear" activation: $a(x) = x$).

use_bias: Boolean, whether the layer uses a bias vector. normalize_weight: Boolean, whether the layer normalizes its complex

weights before convolving the complex input. The complex normalization performed is similar to the one for the batchnorm. Each of the complex kernels is centred and multiplied by the inverse square root of the covariance matrix. Then a complex multiplication is performed as the normalized weights are multiplied by the complex scaling factor gamma.

kernel_initializer: Initializer for the complex *kernel* weights matrix. By default it is 'complex'. The 'complex_independent' and the usual initializers could also be used. (See keras.initializers and init.py).

bias_initializer: Initializer for the bias vector (see keras.initializers).

kernel_regularizer: Regularizer function applied to the *kernel* weights matrix (see keras.regularizers).

bias_regularizer: Regularizer function applied to the bias vector (see keras.regularizers).

activity_regularizer: Regularizer function applied to the output of the layer (its "activation"). (See keras.regularizers).

kernel_constraint: Constraint function applied to the kernel matrix (see keras.constraints).

bias_constraint: Constraint function applied to the bias vector (see keras.constraints).

spectral_parametrization: Boolean, whether or not to use a spectral parametrization of the parameters.

transposed: Boolean, whether or not to use transposed convolution

build(*input_shape*)

call(*inputs*, ***kwargs*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

compute_output_shape(*input_shape*)

Computes the output shape of the layer.

Assumes that the layer will be built to match that input shape provided.

 # Arguments

input_shape: Shape tuple (tuple of integers) or list of shape tuples (one per output tensor of the layer). Shape tuples can include None for free dimensions, instead of an integer.

 # Returns An output shape tuple.

get_config()

Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

class complexnn.conv.**ComplexConv1D**(*filters*, *kernel_size*, *strides=1*, *padding='valid'*, *dilation_rate=1*, *activation=None*, *use_bias=True*, *kernel_initializer='complex'*, *bias_initializer='zeros'*, *kernel_regularizer=None*, *bias_regularizer=None*, *activity_regularizer=None*, *kernel_constraint=None*, *bias_constraint=None*, *seed=None*, *init_criterion='he'*, *spectral_parametrization=False*, *transposed=False*, ***kwargs*)

Bases: *complexnn.conv.ComplexConv*

1D complex convolution layer. This layer creates a complex convolution kernel that is convolved with a complex input layer over a single complex spatial (or temporal) dimension to produce a complex output tensor. If *use_bias* is True, a bias vector is created and added to the complex output. Finally, if *activation* is not *None*, it is applied each of the real and imaginary parts of the output. When using this layer as the first layer in a model, provide an *input_shape* argument (tuple of integers or *None*, e.g. *(10, 128)* for sequences of 10 vectors of 128-dimensional vectors, or *(None, 128)* for variable-length sequences of 128-dimensional vectors. # Arguments

filters: Integer, the dimensionality of the output space, i.e., the number of complex feature maps. It is also the effective number of feature maps for each of the real and imaginary parts. (i.e. the number of complex filters in the convolution) The total effective number of filters is 2 x filters.

kernel_size: An integer or tuple/list of n integers, specifying the dimensions of the convolution window.

strides: An integer or tuple/list of a single integer, specifying the stride length of the convolution. Specifying any stride value != 1 is incompatible with specifying any *dilation_rate* value != 1.

padding: One of *"valid"*, *"causal"* or *"same"* (case-insensitive). *"causal"* results in causal (dilated) convolutions, e.g. output[t] does not depend on input[t+1:]. Useful when modeling temporal data where the model should not violate the temporal order. See [WaveNet: A Generative Model for Raw Audio, section 2.1] (https://arxiv.org/abs/1609.03499).

dilation_rate: an integer or tuple/list of a single integer, specifying the dilation rate to use for dilated convolution. Currently, specifying any *dilation_rate* value != 1 is incompatible with specifying any *strides* value != 1.

activation: Activation function to use (see keras.activations). If you don't specify anything, no activation is applied (ie. "linear" activation: *a(x) = x*).

use_bias: Boolean, whether the layer uses a bias vector. normalize_weight: Boolean, whether the layer normalizes its complex

weights before convolving the complex input. The complex normalization performed is similar to the one for the batchnorm. Each of the complex kernels are centred and multiplied by the inverse square root of covariance matrix. Then, a complex multiplication is perfromed as the normalized weights are multiplied by the complex scaling factor gamma.

kernel_initializer: Initializer for the complex *kernel* weights matrix.

By default it is 'complex'. The 'complex_independent' and the usual initializers could also be used. (see keras.initializers and init.py).

bias_initializer: Initializer for the bias vector (see keras.initializers).

kernel_regularizer: Regularizer function applied to the *kernel* weights matrix (see keras.regularizers).

bias_regularizer: Regularizer function applied to the bias vector (see keras.regularizers).

activity_regularizer: Regularizer function applied to the output of the layer (its "activation"). (see keras.regularizers).

kernel_constraint: Constraint function applied to the kernel matrix (see keras.constraints).

bias_constraint: Constraint function applied to the bias vector (see keras.constraints).

spectral_parametrization: Whether or not to use a spectral parametrization of the parameters.

transposed: Boolean, whether or not to use transposed convolution

 # Input shape 3D tensor with shape: *(batch_size, steps, input_dim)*

 # Output shape 3D tensor with shape: *(batch_size, new_steps, 2 x filters) steps* value might have changed due to padding or strides.

`get_config()`
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

`class complexnn.conv.ComplexConv2D(`*filters*, *kernel_size*, *strides=(1, 1)*, *padding='valid'*, *data_format=None*, *dilation_rate=(1, 1)*, *activation=None*, *use_bias=True*, *kernel_initializer='complex'*, *bias_initializer='zeros'*, *kernel_regularizer=None*, *bias_regularizer=None*, *activity_regularizer=None*, *kernel_constraint=None*, *bias_constraint=None*, *seed=None*, *init_criterion='he'*, *spectral_parametrization=False*, *transposed=False*, ***kwargs*`)`

Bases: `complexnn.conv.ComplexConv`

2D Complex convolution layer (e.g. spatial convolution over images). This layer creates a complex convolution kernel that is convolved with a complex input layer to produce a complex output tensor. If *use_bias* is True, a complex bias vector is created and added to the outputs. Finally, if *activation* is not *None*, it is applied to both the real and imaginary parts of the output. When using this layer as the first layer in a model, provide the keyword argument *input_shape* (tuple of integers, does not include the sample axis), e.g. *input_shape=(128, 128, 3)* for 128x128 RGB pictures in *data_format="channels_last"*. # Arguments

filters: Integer, the dimensionality of the complex output space (i.e, the number complex feature maps in the convolution). The total effective number of filters or feature maps is 2 x filters.

kernel_size: An integer or tuple/list of 2 integers, specifying the width and height of the 2D convolution window. Can be a single integer to specify the same value for all spatial dimensions.

strides: An integer or tuple/list of 2 integers, specifying the strides of the convolution along the width and height. Can be a single integer to specify the same value for all spatial dimensions. Specifying any stride value != 1 is incompatible with specifying any *dilation_rate* value != 1.

padding: one of *"valid"* or *"same"* (case-insensitive). data_format: A string,

one of *channels_last* (default) or *channels_first*. The ordering of the dimensions in the inputs. *channels_last* corresponds to inputs with shape *(batch, height, width, channels)* while *channels_first* corresponds to inputs with shape *(batch, channels, height, width)*. It defaults to the *image_data_format* value found in your Keras config file at *~/.keras/keras.json*. If you never set it, then it will be "channels_last".

dilation_rate: an integer or tuple/list of 2 integers, specifying the dilation rate to use for dilated convolution. Can be a single integer to specify the same value for all spatial dimensions. Currently, specifying any *dilation_rate* value != 1 is incompatible with specifying any stride value != 1.

activation: Activation function to use (see keras.activations). If you don't specify anything, no activation is applied (ie. "linear" activation: *a(x) = x*).

use_bias: Boolean, whether the layer uses a bias vector. normalize_weight: Boolean, whether the layer normalizes its complex

weights before convolving the complex input. The complex normalization performed is similar to the one for the batchnorm. Each of the complex kernels are centred and multiplied by the inverse square root of covariance matrix. Then, a complex multiplication is perfromed as the normalized weights are multiplied by the complex scaling factor gamma.

kernel_initializer: Initializer for the complex *kernel* weights matrix.

By default it is 'complex'. The 'complex_independent' and the usual initializers could also be used. (see keras.initializers and init.py).

bias_initializer: Initializer for the bias vector (see keras.initializers).

kernel_regularizer: Regularizer function applied to the *kernel* weights matrix (see keras.regularizers).

bias_regularizer: Regularizer function applied to the bias vector (see keras.regularizers).

activity_regularizer: Regularizer function applied to the output of the layer (its "activation"). (see keras.regularizers).

kernel_constraint: Constraint function applied to the kernel matrix (see keras.constraints).

bias_constraint: Constraint function applied to the bias vector (see keras.constraints).

spectral_parametrization: Whether or not to use a spectral parametrization of the parameters.

transposed: Boolean, whether or not to use transposed convolution

 # Input shape 4D tensor with shape: *(samples, channels, rows, cols)* if data_format='channels_first' or 4D tensor with shape: *(samples, rows, cols, channels)* if data_format='channels_last'.

 # Output shape 4D tensor with shape: *(samples, 2 x filters, new_rows, new_cols)* if data_format='channels_first' or 4D tensor with shape: *(samples, new_rows, new_cols, 2 x filters)* if data_format='channels_last'. *rows* and *cols* values might have changed due to padding.

`get_config()`
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

class complexnn.conv.**ComplexConv3D**(*filters*, *kernel_size*, *strides=(1, 1, 1)*, *padding='valid'*, *data_format=None*, *dilation_rate=(1, 1, 1)*, *activation=None*, *use_bias=True*, *kernel_initializer='complex'*, *bias_initializer='zeros'*, *kernel_regularizer=None*, *bias_regularizer=None*, *activity_regularizer=None*, *kernel_constraint=None*, *bias_constraint=None*, *seed=None*, *init_criterion='he'*, *spectral_parametrization=False*, *transposed=False*, ***kwargs*)

Bases: *complexnn.conv.ComplexConv*

3D convolution layer (e.g. spatial convolution over volumes). This layer creates a complex convolution kernel that is convolved with a complex layer input to produce a complex output tensor. If *use_bias* is True, a complex bias vector is created and added to the outputs. Finally, if *activation* is not *None*, it is applied to each of the real and imaginary parts of the output. When using this layer as the first layer in a model, provide the keyword argument *input_shape* (tuple of integers, does not include the sample axis), e.g. *input_shape=(2, 128, 128, 128, 3)* for 128x128x128 volumes with 3 channels, in *data_format="channels_last"*. # Arguments

filters: Integer, the dimensionality of the complex output space (i.e, the number complex feature maps in the convolution). The total effective number of filters or feature maps is 2 x filters.

kernel_size: An integer or tuple/list of 3 integers, specifying the width and height of the 3D convolution window. Can be a single integer to specify the same value for all spatial dimensions.

strides: An integer or tuple/list of 3 integers, specifying the strides of the convolution along each spatial dimension. Can be a single integer to specify the same value for all spatial dimensions. Specifying any stride value != 1 is incompatible with specifying any *dilation_rate* value != 1.

padding: one of *"valid"* or *"same"* (case-insensitive). data_format: A string,

one of *channels_last* (default) or *channels_first*. The ordering of the dimensions in the inputs. *channels_last* corresponds to inputs with shape *(batch, spatial_dim1, spatial_dim2, spatial_dim3, channels)* while *channels_first* corresponds to inputs with shape *(batch, channels, spatial_dim1, spatial_dim2, spatial_dim3)*. It defaults to the *image_data_format* value found in your Keras config file at *~/.keras/keras.json*. If you never set it, then it will be "channels_last".

dilation_rate: an integer or tuple/list of 3 integers, specifying the dilation rate to use for dilated convolution. Can be a single integer to specify the same value for all spatial dimensions. Currently, specifying any *dilation_rate* value != 1 is incompatible with specifying any stride value != 1.

activation: Activation function to use (see keras.activations). If you don't specify anything, no activation is applied (ie. "linear" activation: *a(x) = x*).

use_bias: Boolean, whether the layer uses a bias vector. normalize_weight: Boolean, whether the layer normalizes its complex

weights before convolving the complex input. The complex normalization performed is similar to the one for the batchnorm. Each of the complex kernels are centred and multiplied by the inverse square root of covariance matrix. Then, a complex multiplication is perfromed as the normalized weights are multiplied by the complex scaling factor gamma.

kernel_initializer: Initializer for the complex *kernel* weights matrix. By default it is 'complex'. The 'complex_independent' and the usual initializers could also be used. (see keras.initializers and init.py).

bias_initializer: Initializer for the bias vector (see keras.initializers).

kernel_regularizer: Regularizer function applied to the *kernel* weights matrix (see keras.regularizers).

bias_regularizer: Regularizer function applied to the bias vector (see keras.regularizers).

activity_regularizer: Regularizer function applied to the output of the layer (its "activation"). (see keras.regularizers).

kernel_constraint: Constraint function applied to the kernel matrix (see keras.constraints).

bias_constraint: Constraint function applied to the bias vector (see keras.constraints).

spectral_parametrization: Whether or not to use a spectral parametrization of the parameters.

transposed: Boolean, whether or not to use transposed convolution

 # Input shape 5D tensor with shape: *(samples, channels, conv_dim1, conv_dim2, conv_dim3)* if data_format='channels_first' or 5D tensor with shape: *(samples, conv_dim1, conv_dim2, conv_dim3, channels)* if data_format='channels_last'.

 # Output shape 5D tensor with shape: *(samples, 2 x filters, new_conv_dim1, new_conv_dim2, new_conv_dim3)* if data_format='channels_first' or 5D tensor with shape: *(samples, new_conv_dim1, new_conv_dim2, new_conv_dim3, 2 x filters)* if data_format='channels_last'. *new_conv_dim1*, *new_conv_dim2* and *new_conv_dim3* values might have changed due to padding.

get_config()

Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

complexnn.conv.**ComplexConvolution1D**
alias of *complexnn.conv.ComplexConv1D*

complexnn.conv.**ComplexConvolution2D**
alias of *complexnn.conv.ComplexConv2D*

complexnn.conv.**ComplexConvolution3D**
alias of *complexnn.conv.ComplexConv3D*

class complexnn.conv.**WeightNorm_Conv**(*gamma_initializer='ones'*, *gamma_regularizer=None*, *gamma_constraint=None*, *epsilon=1e-07*, ***kwargs*)

Bases: keras.layers.convolutional._Conv

build(*input_shape*)

Creates the layer weights.

Must be implemented on all layers that have weights.

 # Arguments

input_shape: Keras tensor (future input to layer) or list/tuple of Keras tensors to reference for weight shape computations.

call(*inputs*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

get_config()
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

complexnn.conv.**conv2d_transpose**(*inputs*, *filter*, *kernel_size=None*, *filters=None*, *strides=(1, 1)*, *padding='SAME'*, *output_padding=None*, *data_format='channels_last'*)
Compatibility layer for K.conv2d_transpose

Take a filter defined for forward convolution and adjusts it for a transposed convolution.

complexnn.conv.**conv_transpose_output_length**(*input_length*, *filter_size*, *padding*, *stride*, *dilation=1*, *output_padding=None*)
Rearrange arguments for compatibility with conv_output_length.

complexnn.conv.**ifft**(*f*)
Stub

complexnn.conv.**ifft2**(*f*)
Stub

complexnn.conv.**sanitizedInitGet**(*init*)

complexnn.conv.**sanitizedInitSer**(*init*)

complexnn.dense module

class complexnn.dense.**ComplexDense**(*units*, *activation=None*, *use_bias=True*, *init_criterion='he'*, *kernel_initializer='complex'*, *bias_initializer='zeros'*, *kernel_regularizer=None*, *bias_regularizer=None*, *activity_regularizer=None*, *kernel_constraint=None*, *bias_constraint=None*, *seed=None*, ***kwargs*)
Bases: keras.engine.base_layer.Layer

Regular complex densely-connected NN layer. *Dense* implements the operation: *real_preact = dot(real_input, real_kernel) - dot(imag_input, imag_kernel) imag_preact = dot(real_input, imag_kernel) + dot(imag_input, real_kernel) output = activation(K.concatenate([real_preact, imag_preact]) + bias)* where *activation* is the element-wise activation function passed as the *activation* argument, *kernel* is a weights matrix created by the layer, and *bias* is a bias vector created by the layer (only applicable if *use_bias* is *True*). Note: if the input to the layer has a rank greater than 2, then AN ERROR MESSAGE IS PRINTED. # Arguments

units: Positive integer, dimensionality of each of the real part and the imaginary part. It is actualy the number of complex units.

activation: Activation function to use (see keras.activations). If you don't specify anything, no activation is applied (ie. "linear" activation: *a(x) = x*).

use_bias: Boolean, whether the layer uses a bias vector. kernel_initializer: Initializer for the complex *kernel* weights matrix.

By default it is 'complex'. and the usual initializers could also be used. (see keras.initializers and init.py).

bias_initializer: Initializer for the bias vector (see keras.initializers).

kernel_regularizer: Regularizer function applied to the *kernel* weights matrix (see keras.regularizers).

bias_regularizer: Regularizer function applied to the bias vector (see keras.regularizers).

activity_regularizer: Regularizer function applied to the output of the layer (its "activation"). (see keras.regularizers).

kernel_constraint: Constraint function applied to the kernel matrix (see keras.constraints).

bias_constraint: Constraint function applied to the bias vector (see keras.constraints).

 # Input shape a 2D input with shape *(batch_size, input_dim)*.

 # Output shape For a 2D input with shape *(batch_size, input_dim)*, the output would have shape *(batch_size, units)*.

build(*input_shape*)
Creates the layer weights.

Must be implemented on all layers that have weights.

 # Arguments

input_shape: Keras tensor (future input to layer) or list/tuple of Keras tensors to reference for weight shape computations.

call(*inputs*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

compute_output_shape(*input_shape*)
Computes the output shape of the layer.

Assumes that the layer will be built to match that input shape provided.

 # Arguments

input_shape: Shape tuple (tuple of integers) or list of shape tuples (one per output tensor of the layer). Shape tuples can include None for free dimensions, instead of an integer.

 # Returns An output shape tuple.

get_config()
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

complexnn.fft module

class complexnn.fft.**FFT**(***kwargs*)
Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

class complexnn.fft.**FFT2**(***kwargs*)
Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

class complexnn.fft.**IFFT**(***kwargs*)
Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

class complexnn.fft.**IFFT2**(***kwargs*)
Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

complexnn.fft.**fft**(*z*)

complexnn.fft.**fft2**(*x*)

complexnn.fft.**ifft**(*z*)

complexnn.fft.**ifft2**(*x*)

complexnn.init module

class complexnn.init.**ComplexIndependentFilters**(*kernel_size*, *input_dim*, *weight_dim*, *nb_filters=None*, *criterion='glorot'*, *seed=None*)
Bases: keras.initializers.Initializer

get_config()

class complexnn.init.**ComplexInit**(*kernel_size*, *input_dim*, *weight_dim*, *nb_filters=None*, *criterion='glorot'*, *seed=None*)
Bases: keras.initializers.Initializer

class complexnn.init.**IndependentFilters**(*kernel_size*, *input_dim*, *weight_dim*, *nb_filters=None*, *criterion='glorot'*, *seed=None*)
Bases: keras.initializers.Initializer

get_config()

class complexnn.init.**SqrtInit**
Bases: keras.initializers.Initializer

complexnn.init.**complex_init**
alias of *complexnn.init.ComplexInit*

complexnn.init.**independent_filters**
alias of *complexnn.init.IndependentFilters*

complexnn.init.**sqrt_init**
alias of *complexnn.init.SqrtInit*

complexnn.norm module

class complexnn.norm.**ComplexLayerNorm**(*epsilon=0.0001*, *axis=-1*, *center=True*, *scale=True*, *beta_initializer='zeros'*, *gamma_diag_initializer=<function sqrt_init>*, *gamma_off_initializer='zeros'*, *beta_regularizer=None*, *gamma_diag_regularizer=None*, *gamma_off_regularizer=None*, *beta_constraint=None*, *gamma_diag_constraint=None*, *gamma_off_constraint=None*, ***kwargs*)
Bases: keras.engine.base_layer.Layer

build(*input_shape*)
Creates the layer weights.

Must be implemented on all layers that have weights.

 # Arguments

input_shape: Keras tensor (future input to layer) or list/tuple of Keras tensors to reference for weight shape computations.

call(*inputs*)
This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

get_config()
Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

class complexnn.norm.**LayerNormalization**(*epsilon=0.0001*, *axis=-1*, *beta_init='zeros'*, *gamma_init='ones'*, *gamma_regularizer=None*, *beta_regularizer=None*, ***kwargs*)

Bases: keras.engine.base_layer.Layer

build(*input_shape*)

Creates the layer weights.

Must be implemented on all layers that have weights.

 # Arguments

input_shape: Keras tensor (future input to layer) or list/tuple of Keras tensors to reference for weight shape computations.

call(*x*, *mask=None*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

get_config()

Returns the config of the layer.

A layer config is a Python dictionary (serializable) containing the configuration of a layer. The same layer can be reinstantiated later (without its trained weights) from this configuration.

The config of a layer does not include connectivity information, nor the layer class name. These are handled by *Network* (one layer of abstraction above).

 # Returns Python dictionary.

complexnn.norm.**layernorm**(*x*, *axis*, *epsilon*, *gamma*, *beta*)

complexnn.pool module

class complexnn.pool.**SpectralPooling1D**(*topf=(0,)*)

Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

class complexnn.pool.**SpectralPooling2D**(***kwargs*)

Bases: keras.engine.base_layer.Layer

call(*x*, *mask=None*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

complexnn.utils module

class complexnn.utils.**GetAbs**(***kwargs*)

Bases: keras.engine.base_layer.Layer

call(*inputs*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

compute_output_shape(*input_shape*)

Computes the output shape of the layer.

Assumes that the layer will be built to match that input shape provided.

 # Arguments

input_shape: Shape tuple (tuple of integers) or list of shape tuples (one per output tensor of the layer). Shape tuples can include None for free dimensions, instead of an integer.

 # Returns An output shape tuple.

class complexnn.utils.**GetImag**(***kwargs*)

Bases: keras.engine.base_layer.Layer

call(*inputs*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

compute_output_shape(*input_shape*)

Computes the output shape of the layer.

Assumes that the layer will be built to match that input shape provided.

 # Arguments

input_shape: Shape tuple (tuple of integers) or list of shape tuples (one per output tensor of the layer). Shape tuples can include None for free dimensions, instead of an integer.

 # Returns An output shape tuple.

class complexnn.utils.**GetReal**(***kwargs*)

Bases: keras.engine.base_layer.Layer

call(*inputs*)

This is where the layer's logic lives.

 # Arguments inputs: Input tensor, or list/tuple of input tensors. **kwargs: Additional keyword arguments.

 # Returns A tensor or list/tuple of tensors.

compute_output_shape(*input_shape*)

Computes the output shape of the layer.

Assumes that the layer will be built to match that input shape provided.

 # Arguments

input_shape: Shape tuple (tuple of integers) or list of shape tuples (one per output tensor of the layer). Shape tuples can include None for free dimensions, instead of an integer.

 # Returns An output shape tuple.

complexnn.utils.**get_abs**(*x*)

complexnn.utils.**get_imagpart**(*x*)

complexnn.utils.**get_realpart**(*x*)

complexnn.utils.**getpart_output_shape**(*input_shape*)

Module contents

1.4 How to Contribute

1.5 Implementation and Math

Complex convolutional networks provide the benefit of explicitly modelling the phase space of physical systems [TBZ+17]. The complex convolution introduced can be explicitly implemented as convolutions of the real and complex components of both kernels and the data. A complex-valued data matrix in cartesian notation is defined as $\mathbf{M} = M_{\Re} + iM_{\Im}$ and equally, the complex-valued convolutional kernel is defined as $\mathbf{K} = K_{\Re} + iK_{\Im}$. The individual coefficients $(M_{\Re}, M_{\Im}, K_{\Re}, K_{\Im})$ are real-valued matrices, considering vectors are special cases of matrices with one of two dimensions being one.

1.5.1 Complex Convolution Math

The math for complex convolutional networks is similar to real-valued convolutions, with real-valued convolutions being:

$$\int f(y) \cdot g(x-y)\, dy$$

which generalizes to complex-valued function on $\mathbf{R}^d$:

$$(f * g)(x) = \int_{\mathbf{R}^d} f(y)g(x-y)\, dy = \int_{\mathbf{R}^d} f(x-y)g(y)\, dy,$$

in order for the integral to exist, f and g need to decay sufficiently rapidly at infinity [CC-BY-SA Wiki].

1.5.2 Implementation

Solving the convolution of, implemented by [TBZ+17], translated to keras in [DC19]

$$M' = K * M = (M_{\Re} + iM_{\Im}) * (K_{\Re} + iK_{\Im}),$$

we can apply the distributivity of convolutions to obtain

$$M' = \{M_{\Re} * K_{\Re} - M_{\Im} * K_{\Im}\} + i\{M_{\Re} * K_{\Im} + M_{\Im} * K_{\Re}\},$$

where K is the Kernel and M is a data vector.

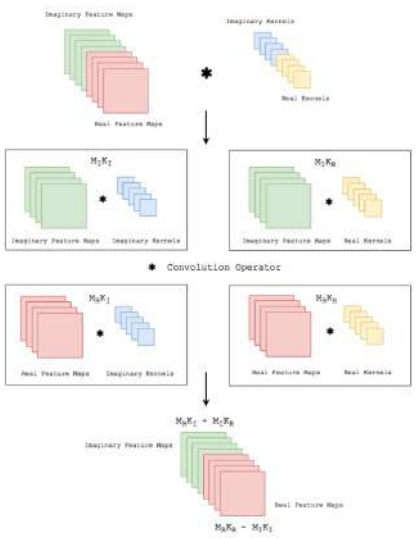


Fig. 1: Complex Convolution implementation (CC-BY [TBZ+17])

1.5.3 Considerations

Complex convolutional neural networks learn by back-propagation. [SSC15] state that the activation functions, as well as the loss function must be complex differentiable (holomorphic). [TBZ+17] suggest that employing complex losses and activation functions is valid for speed, however, refers that [HY12] show that complex-valued networks can be optimized individually with real-valued loss functions and contain piecewise real-valued activations. We reimplement the code [TBZ+17] provides in keras with tensorflow , which provides convenience functions implementing a multitude of real-valued loss functions and activations.

[CC-BY [DLuthjeC19]]

1.6 Citation

Please cite the original work as:

```
@ARTICLE {Trabelsi2017,
    author  = "Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep␣
↪Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio,␣
↪Christopher J Pal",
    title   = "Deep Complex Networks",
    journal = "arXiv preprint arXiv:1705.09792",
    year    = "2017"
}
```

Cite this software version as:

```
@misc{dramsch2019complex,
    title     = {Complex-Valued Neural Networks in Keras with Tensorflow},
    url       = {https://figshare.com/articles/Complex-Valued_Neural_Networks_in_
↪Keras_with_Tensorflow/9783773/1},
    DOI       = {10.6084/m9.figshare.9783773},
    publisher = {figshare},
    author    = {Dramsch, Jesper S{\"o}ren and Contributors},
    year      = {2019}
}
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Bibliography

[DC19] Jesper Soeren Dramsch and Contributors. Complex-valued neural networks in keras with tensorflow. 2019. URL: https://figshare.com/articles/Complex-Valued_Neural_Networks_in_Keras_with_Tensorflow/9783773/1, doi:10.6084/m9.figshare.9783773.

[DLuthjeC19] Jesper Sören Dramsch, Mikael Lüthje, and Anders Nymark Christensen. Complex-valued neural networks for machine learning on non-stationary physical data. *arXiv preprint arXiv:1905.12321*, 2019.

[HY12] Akira Hirose and Shotaro Yoshida. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. *IEEE Transactions on Neural Networks and Learning Systems*, 2012.

[SSC15] Andy M. Sarroff, Victor Shepardson, and Michael A. Casey. Learning representations using complex-valued nets. *CoRR*, 2015. URL: http://arxiv.org/abs/1511.06351, arXiv:1511.06351.

[TBZ+17] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2017.

Python Module Index

C

Index

Acronyms

AE AutoEncoder . . . 18
AI Artificial Intelligence . . . 7
API Application Programming Interface . . . 8
ASI Automatic Seismic Interpretation . . . 21
ANN Artificial Neural Network . . . 9
BHI Borehole Imaging
BN Batch Normalization . . . 11
BSEM Backscatter Scanning-Electron Microscopy . . . 27
CNN Convolutional Neural Network . . . 8
CMYK Cyan-Magenta-Yellow-blacK . . . 26
CPU Central Processing Unit . . . 11
DCGAN Deep Convolutional Generative Adversarial Network . . . 15
DFN Discrete Fracture Network . . . 22
DL Deep Learning . . . 6
DNN Deep Neural Network . . . 8
DT Decision Tree . . . 8
DTW Dynamic Time Warping . . . 5
ELU Expenantial Linear Unit
FCN Fully Convolutional Network . . . 18
FK Frequency-Wavenumber . . . 32
FGPA Field Programmable Gate Array . . . 11
FFT Fast Fourier transform . . . 15
FOSS Free Open Source Software
GAN Generative Adversarial Network . . . 9
GMM Gaussian mixture model . . . 60
GPR Ground Penetrating Radar . . . 21
GPU Graphical Processing Unit . . . 11

HBR Hatchell-Bourne-Røste . . . 4
HMM Hidden Markov Model . . . 20
HSV Hue-Saturation-Value . . . 26
iid independent and identically distributed . . . 22
KL Kullback-Leibler
KNN k-Nearest Neighbour . . . 8
LiME local interpretable model-agnostic explanation . . . 24
LSTM Long Short-Term Memory . . . 9
MAE Mean Absolute Error . . . 11
ML Machine Learning . . . 5
MLP Multi-Layer Perceptron . . . 9
MSE Mean Squared Error . . . 11
NAS Neural Architecture Search . . . 17
NLP Natural Language Processing . . . 9
NN Neural Network . . . 3
NRMS Normalized Root Mean Squared Error . . . 4
PReLU Parameterized Rectified Linear Unit
QI Qantitative Interpretation . . . 5
ReLU Rectified Linear Unit
RF Random Forest . . . 8
RGB Red-Green-Blue . . . 26
RL Reinforcement Learning
RNN Recurrent Neural Network . . . 8
RMM Random Markov Model . . . 20
SGD Stochastic Gradient Descent . . . 11
SOM Self-Organizing Map . . . 13
SOTA state-of-the-art
SVM Support Vector Machine . . . 8
TF Tensorflow . . . 11
TPU Tensor Processing Unit . . . 11
TPE Tree of Parzen Estimator . . . 35
VAE Variational AutoEncoder . . . 21
VSP Vertical Seismic Profiling

Bibliography

Aabø, T. M., J. S. Dramsch, M. Welch, and M. Lüthje (2017). "Correlation of Fractures From Core, Borehole Images and Seismic Data in a Chalk Reservoir in the Danish North Sea". In: *79th EAGE Conference and Exhibition 2017*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201701283. URL: https://doi.org/10.3997/2214-4609.201701283 (cit. on pp. 25, 28, 38, 41, 42).

Aabø, T. M., J. S. Dramsch, C. L. Würtzen, S. Seyum, F. Amour, M. Welch, and M. Lüthje (2020). "An integrated workflow for fracture characterization in chalk reservoirs, applied to the Kraka Field". In: *Marine and Petroleum Geology* 112. Published, Chapter 4. ISSN: 0264-8172. DOI: https://doi.org/10.1016/j.marpetgeo.2019.104065. URL: http://www.sciencedirect.com/science/article/pii/S026481721930501X (cit. on pp. 25, 26, 28, 38, 41, 47).

Alikhassi, A., H. E. Gourabi, and M. Baikpour (2018). "Comparison of inter-and intra-observer variability of breast density assessments using the fourth and fifth editions of Breast Imaging Reporting and Data System". In: *European journal of radiology open* 5, pp. 67–72 (cit. on p. 23).

AlRegib, G., M. Deriche, Z. Long, H. Di, Z. Wang, Y. Alaudah, M. A. Shafiq, and M. Alfarraj (2018). "Subsurface Structure Analysis Using Computational Interpretation and Learning: A Visual Signal Processing Perspective". In: *IEEE Signal Process. Mag.* 35.2, pp. 82–98. ISSN: 1053-5888. DOI: 10.1109/MSP.2017.2785979. URL: http://dx.doi.org/10.1109/MSP.2017.2785979 (cit. on p. 134).

Anifowose, F., C. Ayadiuno, and F. Rashedian (2017a). "Carbonate Reservoir Cementation Factor Modeling Using Wireline Logs and Artificial Intelligence Methodology". In: *79th EAGE Conference and Exhibition 2017-Workshops* (cit. on p. 21).

Anifowose, F., C. Ayadiuno, and F. Rashedian (2017b). "Carbonate Reservoir Cementation Factor Modeling Using Wireline Logs and Artificial Intelligence Methodology". In: *79th EAGE Conference and Exhibition 2017-Workshops*. earthdoc.org. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=89285 (cit. on p. 134).

Ansari, H. R. (2014). "Use seismic colored inversion and power law committee machine based on imperial competitive algorithm for improving porosity prediction in a heterogeneous reservoir". In: *J. Appl. Geophys.* 108, pp. 61–68. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2014.06.016. URL: http://www.sciencedirect.com/science/article/pii/S092698511400192X (cit. on p. 134).

Araya-Polo, M., T. Dahlke, C. Frogner, C. Zhang, T. Poggio, and D. Hohl (2017). "Automated fault detection without seismic processing". In: *Lead. Edge* 36.3, pp. 208–214.

ISSN: 1070-485X. DOI: 10.1190/tle36030208.1. URL: https://doi.org/10.1190/tle36030208.1 (cit. on p. 134).

Araya-Polo, M., J. Jennings, A. Adler, and T. Dahlke (2018). "Deep-learning tomography". In: *Lead. Edge* 37.1, pp. 58–66. ISSN: 1070-485X. DOI: 10.1190/tle37010058.1. URL: https://doi.org/10.1190/tle37010058.1 (cit. on pp. 21, 134).

Arjovsky, M., S. Chintala, and L. Bottou (2017). "Wasserstein gan". In: *arXiv preprint arXiv:1701.07875* (cit. on p. 15).

Arts, R., O. Eiken, A. Chadwick, P. Zweigel, L. van der Meer, and B. Zinszner (2004). "Monitoring of CO2 injected at Sleipner using time-lapse seismic data". In: *Energy* 29.9-10, pp. 1383–1392. ISSN: 0360-5442. DOI: 10.1016/j.energy.2004.03.072. URL: http://www.sciencedirect.com/science/article/pii/S0360544204001550 (cit. on p. 3).

Ashida, Y. (1996). "Data processing of reflection seismic data by use of neural network". In: *J. Appl. Geophys.* 35.2, pp. 89–98. ISSN: 0926-9851. DOI: 10.1016/0926-9851(96)00010-9. URL: http://www.sciencedirect.com/science/article/pii/0926985196000109 (cit. on p. 134).

Asoodeh, M. and P. Bagheripour (2014). "ACE stimulated neural network for shear wave velocity determination from well logs". In: *J. Appl. Geophys.* 107, pp. 102–107. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2014.05.014. URL: http://www.sciencedirect.com/science/article/pii/S0926985114001487 (cit. on p. 134).

Baan, M. van der and C. Jutten (2000). "Neural networks in geophysical applications". In: *Geophysics* 65.4, pp. 1032–1047. ISSN: 0016-8033. DOI: 10.1190/1.1444797. URL: https://doi.org/10.1190/1.1444797 (cit. on p. 19).

Bagheripour, P. (2014). "Committee neural network model for rock permeability prediction". In: *J. Appl. Geophys.* 104, pp. 142–148. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2014.03.001. URL: http://www.sciencedirect.com/science/article/pii/S092698511400069X (cit. on p. 133).

Ballabio, C. and S. Sterlacchini (2012). "Support Vector Machines for Landslide Susceptibility Mapping: The Staffora River Basin Case Study, Italy". In: *Math. Geosci.* 44.1, pp. 47–70. ISSN: 1874-8961, 1874-8953. DOI: 10.1007/s11004-011-9379-9. URL: https://doi.org/10.1007/s11004-011-9379-9 (cit. on p. 21).

Bauer, K., R. G. Pratt, C. Haberland, and M. Weber (2008). "Neural network analysis of crosshole tomographic images: The seismic signature of gas hydrate bearing sediments in the Mackenzie Delta (NW Canada)". In: *Geophys. Res. Lett.* 35.19, p. 340. ISSN: 0094-8276. DOI: 10.1029/2008GL035263. URL: http://doi.wiley.com/10.1029/2008GL035263 (cit. on p. 134).

Bauer, K., J. Kulenkampff, J. Henninges, and E. Spangenberg (2015). "Lithological control on gas hydrate saturation as revealed by signal classification of NMR logging data". In: *J. Geophys. Res. [Solid Earth]* 120.9, pp. 6001–6017. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2015JB012150 (cit. on p. 134).

Bayes, T. (1763). "LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, FRS communicated by Mr. Price, in a letter to John Canton, AMFR S". In: *Philosophical transactions of the Royal Society of London* 53, pp. 370–418 (cit. on p. 7).

Belson, W. A. (1959). "Matching and prediction on the principle of biological classification". In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 8.2, pp. 65–75 (cit. on p. 8).

Bennett, J., S. Lanning, et al. (2007). "The netflix prize". In: *Proceedings of KDD cup and workshop*. Vol. 2007. New York, NY, USA., p. 35 (cit. on p. 8).

Bergstra, J., B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox (2015). "Hyperopt: a python library for model selection and hyperparameter optimization". In: *Computational Science & Discovery* 8.1, p. 014008 (cit. on p. 35).

Bestagini, P., V. Lipari, and S. Tubaro (2017). "A machine learning approach to facies classification using well logs". In: *SEG Technical Program Expanded Abstracts 2017*. SEG Technical Program Expanded Abstracts. Society of Exploration Geophysicists, pp. 2137–2142. DOI: 10.1190/segam2017-17729805.1. URL: https://doi.org/10.1190/segam2017-17729805.1 (cit. on p. 21).

Beyreuther, M. and J. Wassermann (2008). "Continuous earthquake detection and classification using discrete Hidden Markov Models". In: *Geophys. J. Int.* 175.3, pp. 1055–1066. ISSN: 0956-540X. DOI: 10.1111/j.1365-246X.2008.03921.x. URL: https://academic.oup.com/gji/article-abstract/175/3/1055/634811 (cit. on p. 22).

Bhatt, A. and H. B. Helle (2002). "Determination of facies from well logs using modular neural networks". In: *Pet. Geosci.* ISSN: 1354-0793. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=37980 (cit. on p. 134).

Bhowmick, D., D. K. Gupta, S. Maiti, and U. Shankar (2018). "Deep Autoassociative Neural Networks for Noise Reduction in Seismic data". In: *ArXiv*. arXiv: 1805.00291 [cs.CE]. URL: http://arxiv.org/abs/1805.00291 (cit. on p. 134).

Bicego, M., C. Acosta-Muñoz, and M. Orozco-Alzate (2013). "Classification of Seismic Volcanic Signals Using Hidden-Markov-Model-Based Generative Embeddings". In: *IEEE Trans. Geosci. Remote Sens.* 51.6, pp. 3400–3409. ISSN: 0196-2892. DOI: 10.1109/TGRS.2012.2220370. URL: http://dx.doi.org/10.1109/TGRS.2012.2220370 (cit. on p. 22).

Birkenfeld, S. (2010). "Automatic detection of reflexion hyperbolas in GPR data with neural networks". In: *World Automation Congress*. researchgate.net, pp. 1–6. URL: https://www.researchgate.net/profile/Sven_Birkenfeld/publication/221671913_Automatic_detection_of_reflexion_hyperbolas_in_GPR_data_with_neural_networks/links/00b7d5303634cb6e1a000000.pdf (cit. on p. 133).

Bishop, C. M. (1995). "Training with noise is equivalent to Tikhonov regularization". In: *Neural computation* 7.1, pp. 108–116 (cit. on p. 35).

Bishop, C. M. (2016). *Pattern Recognition and Machine Learning*. en. Springer New York. ISBN: 9781493938438. URL: https://market.android.com/details?id=book-kOXDtAEACAAJ (cit. on p. 6).

Blouin, M., A. Caté, L. Perozzi, and E. Gloaguen (2017). "Automated facies prediction in drillholes using machine learning". In: *79th EAGE Conference and Exhibition 2017-Workshops*. earthdoc.org. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=89276 (cit. on p. 21).

Boateng, C., L. Fu, W. Yu, and G. Xizhu (2017). "Porosity inversion by Caianiello neural networks with Levenberg-Marquardt optimization". In: *Interpretation* 5.3, SL33–

SL42. ISSN: 2324-8858. DOI: 10.1190/INT-2016-0119.1. URL: https://doi.org/10.1190/INT-2016-0119.1 (cit. on p. 133).

Braeuer, B. and K. Bauer (2015). "A new interpretation of seismic tomography in the southern Dead Sea basin using neural network clustering techniques: INTERPRETATION OF TOMOGRAPHY IN THE SDSB". In: *Geophys. Res. Lett.* Lecture Notes Comput. Sci 42.22, pp. 9772–9780. ISSN: 0094-8276. DOI: 10.1002/2015GL066559. URL: http://doi.wiley.com/10.1002/2015GL066559 (cit. on p. 134).

Brown, T. B., D. Mané, A. Roy, M. Abadi, and J. Gilmer (2017). "Adversarial patch". In: *arXiv preprint arXiv:1712.09665* (cit. on p. 23).

Bruges: Bag of really useful geophysical equations and stuff (2016). URL: https://github.com/agile-geoscience/bruges (cit. on p. xii).

Bryson, A. E. (1961). "A gradient method for optimizing multi-stage allocation processes". In: *Proc. Harvard Univ. Symposium on digital computers and their applications*. Vol. 72 (cit. on p. 8).

Buitinck, L., G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux (2013). "API design for machine learning software: experiences from the scikit-learn project". In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122 (cit. on pp. 8, 21).

Calderón-Macías, C., M. Sen, and P. Stoffa (1997). "Hopfield neural networks, and mean field annealing for seismic deconvolution and multiple attenuation". In: *Geophysics* 62.3, pp. 992–1002. ISSN: 0016-8033. DOI: 10.1190/1.1444205. URL: https://doi.org/10.1190/1.1444205 (cit. on p. 133).

Cao, J. and B. Roy (2017). "Time-lapse reservoir property change estimation from seismic using machine learning". In: *Lead. Edge* 36.3, pp. 234–238. ISSN: 1070-485X. DOI: 10.1190/tle36030234.1. URL: https://doi.org/10.1190/tle36030234.1 (cit. on p. 22).

Carreira, V., C. P. Neto, and R. Bijani (2018). "A Comparison of Machine Learning Processes for Classification of Rock Units Using Well Log Data". In: *80th EAGE Conference and Exhibition 2018*. earthdoc.org. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92908 (cit. on p. 134).

Castellaro, S. and F. Mulargia (2007). "Classification of pre-eruption and non-pre-eruption epochs at Mount Etna volcano by means of artificial neural networks". In: *Geophys. Res. Lett.* 34.10. ISSN: 0094-8276. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2007GL029513 (cit. on p. 134).

Caté, A., L. Perozzi, E. Gloaguen, and M. Blouin (2017). "Machine learning as a tool for geologists". In: *Lead. Edge* 36.3, pp. 215–219. ISSN: 1070-485X. DOI: 10.1190/tle36030215.1. URL: https://doi.org/10.1190/tle36030215.1 (cit. on p. 22).

Caté, A., E. Schetselaar, P. Mercier-Langevin, and P.-S. Ross (2018). "Classification of lithostratigraphic and alteration units from drillhole lithogeochemical data using machine learning: A case study from the Lalor volcanogenic massive sulphide deposit, Snow Lake, Manitoba, Canada". In: *J. Geochem. Explor.* 188, pp. 216–228. ISSN: 0375-6742. URL: https://www.sciencedirect.com/science/article/pii/S0375674217305083 (cit. on pp. 21, 22).

Chaki, S., A. Routray, and W. K. Mohanty (2018). "Well-Log and Seismic Data Integration for Reservoir Characterization: A Signal Processing and Machine-Learning Perspective". In: *IEEE Signal Process. Mag.* 35.2, pp. 72–81. ISSN: 1053-5888. DOI: 10.1109/MSP.2017.2776602. URL: http://dx.doi.org/10.1109/MSP.2017.2776602 (cit. on pp. 21, 134).

Chang, C.-C. and C.-J. Lin (2011). "LIBSVM: A Library for Support Vector Machines". In: *ACM Trans. Intell. Syst. Technol.* 2.3, 27:1–27:27. ISSN: 2157-6904. DOI: 10.1145/1961189.1961199. URL: http://doi.acm.org/10.1145/1961189.1961199 (cit. on p. 8).

Chang, H.-C., D. C. Kopaska-Merkel, and H.-C. Chen (2002). "Identification of lithofacies using Kohonen self-organizing maps". In: *Comput. Geosci.* 28.2, pp. 223–229. ISSN: 0098-3004. DOI: 10.1016/S0098-3004(01)00067-X. URL: http://www.sciencedirect.com/science/article/pii/S009830040100067X (cit. on p. 134).

Chen, T. and C. Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785 (cit. on p. 8).

Chen, Y. (2015). "Mineral potential mapping with a restricted Boltzmann machine". In: *Ore Geol. Rev.* 71, pp. 749–760. ISSN: 0169-1368. DOI: 10.1016/j.oregeorev.2014.08.012. URL: http://www.sciencedirect.com/science/article/pii/S0169136814002029 (cit. on p. 133).

Chen, Y., L. Lu, and X. Li (2014). "Application of continuous restricted Boltzmann machine to identify multivariate geochemical anomaly". In: *J. Geochem. Explor.* 140, pp. 56–63. ISSN: 0375-6742. DOI: 10.1016/j.gexplo.2014.02.013. URL: http://www.sciencedirect.com/science/article/pii/S0375674214000764 (cit. on p. 133).

Chen, Y., H. Fang, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, S. Yan, and J. Feng (2019). "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution". In: *arXiv preprint arXiv:1904.05049* (cit. on p. 131).

Chevitarese, D., D. Szwarcman, R. M. D. Silva, and E. V. Brazil (2018). "Seismic facies segmentation using deep learning". In: *ACE 2018 Annual Convention & Exhibition*. searchanddiscovery.com. URL: http://www.searchanddiscovery.com/documents/2018/42286chevitarese/ndx_chevitarese.pdf (cit. on p. 134).

Ching, T., D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman, W. Xie, G. L. Rosen, B. J. Lengerich, J. Israeli, J. Lanchantin, S. Woloszynek, A. E. Carpenter, A. Shrikumar, J. Xu, E. M. Cofer, C. A. Lavender, S. C. Turaga, A. M. Alexandari, Z. Lu, D. J. Harris, D. DeCaprio, Y. Qi, A. Kundaje, Y. Peng, L. K. Wiley, M. H. S. Segler, S. M. Boca, S. J. Swamidass, A. Huang, A. Gitter, and C. S. Greene (2018). "Opportunities and obstacles for deep learning in biology and medicine". en. In: *J. R. Soc. Interface* 15.141. ISSN: 1742-5689, 1742-5662. DOI: 10.1098/rsif.2017.0387. URL: http://dx.doi.org/10.1098/rsif.2017.0387 (cit. on p. 6).

Chollet, F. et al. (2015). *Keras*. https://keras.io (cit. on p. 6).

Collobert, R., S. Bengio, and J. Mariéthoz (2002). *Torch: a modular machine learning software library*. Tech. rep. Idiap (cit. on p. 8).

Corte, G., J. S. Dramsch, C. MacBeth, and H. Amini (2019). "Exploring training possibilities in a Deep Neural network application for Inverting 4D seismic maps to changes in pressure and saturation". In: *TBD*. In Preparation, Not Included (cit. on pp. 36, 39).

Cortes, C. and V. Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297 (cit. on p. 8).

Cover, T. and P. Hart (1967). "Nearest neighbor pattern classification". In: *IEEE transactions on information theory* 13.1, pp. 21–27 (cit. on p. 8).

Cui, Y.-A., L. Wang, and J.-P. Xiao (2010). "Automatic feature recognition for GPR image processing". In: *Proc. World Acad. of Sci. Eng. Technol.* 61, pp. 176–179. ISSN: 1307-6884. URL: https://pdfs.semanticscholar.org/edd6/3447f33c032fe26dfb970e92f6194e98d pdf (cit. on p. 133).

Dai, H. and C. MacBeth (1994). "A Neural network picker for VSP data". In: *56th EAEG Meeting*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=12499 (cit. on p. 134).

Dai, H. and C. MacBeth (1997a). "The application of back-propagation neural network to automatic picking seismic arrivals from single-component recordings". In: *J. Geophys. Res. [Solid Earth]*. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/97JB00625 (cit. on p. 133).

Dai, H. and C. MacBeth (1995). "Automatic picking of seismic arrivals in local earthquake data using an artificial neural network". In: *Geophys. J. Int.* 120.3, pp. 758–774. ISSN: 0956-540X. DOI: 10.1111/j.1365-246X.1995.tb01851.x. URL: https://academic.oup.com/gji/article-abstract/120/3/758/779585 (cit. on p. 133).

Dai, H. and C. MacBeth (1997b). "Application of back-propagation neural networks to identification of seismic arrival types". In: *Phys. Earth Planet. Inter.* 101.3, pp. 177–188. ISSN: 0031-9201. DOI: 10.1016/S0031-9201(97)00004-6. URL: http://www.sciencedirect.com/science/article/pii/S0031920197000046 (cit. on p. 134).

Dammeier, F., J. R. Moore, C. Hammer, F. Haslinger, and S. Loew (2016). "Automatic detection of alpine rockslides in continuous seismic data using hidden Markov models". In: *J. Geophys. Res. Earth Surf.* of the Ser. Lect. Notes in Comput. Sci 121.2, pp. 351–371. ISSN: 2169-9003. DOI: 10.1002/2015JF003647. URL: http://doi.wiley.com/10.1002/2015JF003647 (cit. on p. 22).

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255 (cit. on p. 8).

DeVries, P. M., F. Viégas, M. Wattenberg, and B. J. Meade (2018). "Deep learning of aftershock patterns following large earthquakes". In: *Nature* 560.7720, p. 632 (cit. on p. 21).

Di, H., M. Shafiq, and G. AlRegib (2017a). "Multi-attribute k-means cluster analysis for salt boundary detection". In: *79th EAGE Conference and Exhibition*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=88632 (cit. on p. 22).

Di, H., M. A. Shafiq, and G. AlRegib (2017b). "Seismic-fault detection based on multiattribute support vector machine analysis". In: *SEG Technical Program Expanded Abstracts 2017*. Society of Exploration Geophysicists, pp. 2039–2044 (cit. on p. 21).

Dodge, D. A. and D. B. Harris (2016). "Large-scale test of dynamic correlation processors: Implications for correlation-based seismic pipelines". In: *Bull. Seismol. Soc. Am.* URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/106/2/435/332173 (cit. on p. 21).

Dorigo, M. (1992). "Optimization, learning and natural algorithms". In: *PhD Thesis, Politecnico di Milano* (cit. on p. 26).

Dowla, F. U., S. R. Taylor, and R. W. Anderson (1990). "Seismic discrimination with artificial neural networks: Preliminary results with regional spectral data". In: *Bull. Seismol. Soc. Am.* 80.5, pp. 1346–1373. ISSN: 0037-1106. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/80/5/1346/119382 (cit. on p. 134).

Draelos, T. J., S. Ballard, C. J. Young, et al. (2015). "A new method for producing automated seismic bulletins: Probabilistic event detection, association, and location". In: *Bulletin of the*. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/105/5/2453/331946 (cit. on p. 134).

Dramsch, J. S. (2018). *Reproducible Code: Deep-learning seismic facies on state-of-the-art CNN architectures*. DOI: 10.6084/m9.figshare.7227545. URL: https://github.com/JesperDramsch/segam18 (cit. on p. 39).

Dramsch, J. S. (2019). *Reproducible Code: Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks*. URL: https://github.com/JesperDramsch/voxelmorph-seismic (cit. on p. 39).

Dramsch, J. S. (2020). *Reproducible Code: Dynamic Timewarping Tutorial – Geophysics*. URL: https://github.com/JesperDramsch/dtw-tutorial (cit. on p. 39).

Dramsch, J. S., F. Amour, and M. Lüthje (2018a). "Gaussian Mixture Models For Robust Unsupervised Scanning-Electron Microscopy Image Segmentation Of North Sea Chalk". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 4. EAGE. DOI: 10.3997/2214-4609.201803014. URL: https://doi.org/10.3997/2214-4609.201803014 (cit. on pp. 25, 27, 38, 41, 60).

Dramsch, J. S., A. N. Christensen, and M. Lüthje (2019a). "Let's do the Time Warp again! – Revisiting Dynamic Time Warping – A practical tutorial in Python on North Sea field data". In: *Geophysics*. In Review, Chapter 5 (cit. on pp. 28, 29, 31, 39, 63, 64).

Dramsch, J. S., A. N. Christensen, C. MacBeth, and M. Lüthje (2019b). "Deep Unsupervised 4D Seismic 3D Time-Shift Estimation with Convolutional Neural Networks". In: *IEEE Transactions in Geoscience and Remote Sensing*. Submitted, Chapter 7 (cit. on pp. 23, 36, 37, 39, 117).

Dramsch, J. S. and Contributors (2018b). *Awesome Open Geoscience*. Maintainer. URL: https://github.com/softwareunderground/awesome-open-geoscience (cit. on p. xii).

Dramsch, J. S. and Contributors (2019c). *Complex-Valued Neural Networks in Keras with Tensorflow*. Open-Source Software. DOI: 10.6084/m9.figshare.9783773. URL: https://github.com/JesperDramsch/keras-complex (cit. on pp. 38, 135).

Dramsch, J. S., G. Corte, H. Amini, M. Lüthje, and C. MacBeth (2019d). "Deep Learning Application for 4D Pressure Saturation Inversion Compared to Bayesian Inversion on North Sea Data". In: *Second EAGE Workshop Practical Reservoir Monitoring 2019*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201900028 (cit. on pp. 34, 35, 39, 105, 111).

Dramsch, J. S., G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019e). "Including Physics in Deep Learning – An Example from 4D Seismic Pressure Saturation Inversion". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme*. Published, Chapter 6. EAGE. DOI: 10.3997/2214-4609.201901967. URL: https://doi.org/10.3997/2214-4609.201901967 (cit. on pp. 34, 39, 105, 106).

Dramsch, J. S., G. Corte, H. Amini, C. MacBeth, and M. Lüthje (2019f). "Physics-based deep neural encoders-decoder network for 4D seismic pressure-saturation inversion on North Sea data". In: *TBD*. In Preparation, Not Included (cit. on pp. 36, 39).

Dramsch, J. S. and M. Lüthje (2018c). "Deep-learning seismic facies on state-of-the-art CNN architectures". In: *SEG Technical Program Expanded Abstracts 2018*. Published, Chapter 5. Society of Exploration Geophysicists, pp. 2036–2040. DOI: 10.1190/segam2018-2996783.1. URL: https://doi.org/10.1190/segam2018-2996783.1 (cit. on pp. 17, 23, 33, 38, 63, 99, 134).

Dramsch, J. S. and M. Lüthje (2018d). "Information Theory Considerations In Patch-Based Training Of Deep Neural Networks On Seismic Time-Series". In: *First EAGE/PESGB Workshop Machine Learning*. Published, Chapter 5. EAGE. DOI: 10.3997/2214-4609.201803020. URL: https://doi.org/10.3997/2214-4609.201803020 (cit. on pp. 31, 32, 38, 63, 96).

Dramsch, J. S., M. Lüthje, and A. N. Christensen (2019g). "Complex-valued neural networks for machine learning on non-stationary physical data". In: *Computers & Geoscience*. Submitted, Chapter 5 (cit. on pp. 31, 32, 38, 63, 74).

Dreyfus, S. (1962). "The numerical solution of variational problems". In: *Journal of Mathematical Analysis and Applications* 5.1, pp. 30–45 (cit. on p. 8).

Emelyanova, I., M. Pervukhina, M. Clennell, et al. (2017). "Unsupervised identification of electrofacies employing machine learning". In: *79th EAGE Conference*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=89274 (cit. on p. 134).

Esposito, A. M., F. Giudicepietro, S. Scarpetta, L. D'Auria, M. Marinaro, and M. Martini (2006). "Automatic Discrimination among Landslide, Explosion-Quake, and Microtremor Seismic Signals at Stromboli Volcano Using Neural Networks". In: *Bull. Seismol. Soc. Am.* 96.4A, pp. 1230–1240. ISSN: 0037-1106. DOI: 10.1785/0120050097. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/96/4A/1230/146685 (cit. on p. 134).

Falsaperla, S., S. Graziani, G. Nunnari, and S. Spampinato (1996). "Automatic classification of volcanic earthquakes by using Multi-Layered neural networks". In: *Nat.*

Hazards 13.3, pp. 205–228. ISSN: 0921-030X, 1573-0840. DOI: 10.1007/BF00215816. URL: https://doi.org/10.1007/BF00215816 (cit. on p. 134).

Feng, X.-T. and M. Seto (1998). "Neural network dynamic modelling of rock microfracturing sequences under triaxial compressive stress conditions". In: *Tectonophysics* 292.3, pp. 293–309. ISSN: 0040-1951. DOI: 10.1016/S0040-1951(98)00072-9. URL: http://www.sciencedirect.com/science/article/pii/S0040195198000729 (cit. on p. 20).

Ferreira, R., E. V. Brazil, R. Silva, et al. (2018). "Texture-Based Similarity Graph to Aid Seismic Interpretation". In: *ACE 2018 Annual*. URL: http://www.searchanddiscovery.com/documents/2018/70365ferreira/ndx_ferreira.pdf (cit. on p. 22).

Forel, D., T. Benz, and W. D. Pennington (2005). *Seismic Data Processing with Seismic Un∗ x: A 2D Seismic Data Processing Primer*. Society of Exploration Geophysicists (cit. on p. 23).

Fukushima, K. (1980). "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position". In: *Biological cybernetics* 36.4, pp. 193–202 (cit. on p. 8).

Fung, C. C., K. W. Wong, and H. Eren (1997). "Modular artificial neural network for prediction of petrophysical properties from well log data". In: *IEEE Trans. Instrum. Meas.* 46.6, pp. 1295–1299. ISSN: 0018-9456. DOI: 10.1109/19.668276. URL: http://dx.doi.org/10.1109/19.668276 (cit. on p. 134).

Gamba, P. and S. Lossani (2000). "Neural detection of pipe signatures in ground penetrating radar images". In: *IEEE Trans. Geosci. Remote Sens.* 38.2, pp. 790–797. ISSN: 0196-2892. DOI: 10.1109/36.842008. URL: http://dx.doi.org/10.1109/36.842008 (cit. on p. 133).

Gentili, S. and A. Michelini (2006). "Automatic picking of P and S phases using a neural tree". In: *J. Seismol.* 10.1, pp. 39–63. ISSN: 1383-4649, 1573-157X. DOI: 10.1007/s10950-006-2296-6. URL: https://doi.org/10.1007/s10950-006-2296-6 (cit. on p. 134).

Ghaderi, A. and M. Landrø (2005). "Pre-stack estimation of time-lapse seismic velocity changes—an example from the Sleipner CO2-sequestration project". In: *Greenhouse Gas Control Technologies 7*. Elsevier, pp. 633–641 (cit. on p. 5).

Glorot, X. and Y. Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256 (cit. on p. 10).

Goldberg, D. E. and J. H. Holland (1988). "Genetic Algorithms and Machine Learning". en. In: *Mach. Learn.* 3.2-3, pp. 95–99. ISSN: 0885-6125, 1573-0565. DOI: 10.1023/A:1022602019183. URL: https://link.springer.com/article/10.1023/A:1022602019183 (cit. on p. 7).

Golsanami, N., A. Kadkhodaie-Ilkhchi, and A. Erfani (2015). "Synthesis of capillary pressure curves from post-stack seismic data with the use of intelligent estimators: a case study from the Iranian part of the South Pars …". In: *Journal of Applied*. URL: https://www.sciencedirect.com/science/article/pii/S0926985114003413 (cit. on p. 134).

Goodfellow, I. J., J. Shlens, and C. Szegedy (2014a). "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (cit. on p. 23).

Goodfellow, I. J., D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, et al. (2013). "Challenges in representation learning: A report on three machine learning contests". In: *International Conference on Neural Information Processing*. Springer, pp. 117–124 (cit. on p. 8).

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press (cit. on p. 11).

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014b). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Curran Associates, Inc., pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf (cit. on p. 15).

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014c). "Generative adversarial nets". In: *Advances in neural information processing systems*, pp. 2672–2680 (cit. on p. 9).

Gramstad, O. and M. Nickel (2018). "Automated Top Salt Interpretation Using a Deep Convolutional Net". In: *80th EAGE Conference and Exhibition 2018*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92117 (cit. on p. 134).

Guillen, P., G. Larrazabal*, G. González, D. Boumber, and R. Vilalta (2015). "Supervised learning to detect salt body". In: *SEG Technical Program Expanded Abstracts 2015*. SEG Technical Program Expanded Abstracts. Society of Exploration Geophysicists, pp. 1826–1829. DOI: 10.1190/segam2015-5931401.1. URL: https://doi.org/10.1190/segam2015-5931401.1 (cit. on p. 21).

Guitton, A. (2018). "3D Convolutional Neural Networks for Fault Interpretation". In: *80th EAGE Conference and Exhibition 2018*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92118 (cit. on p. 134).

Gulrajani, I., F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville (2017). "Improved training of wasserstein gans". In: *Advances in neural information processing systems*, pp. 5767–5777 (cit. on p. 15).

Guo, R., Y. S. Zhang, H. Lin, and W. Liu (2017). "Sweet Spot Interpretation from Multiple Attributes: Machine Learning and Neural Networks Technologies". In: *First EAGE/AMGP/AMGE Latin*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=90731 (cit. on p. 134).

Gupta, I., C. Rai, C. Sondergeld, and D. Devegowda (2018). "Rock typing in the Upper Devonian-Lower Mississippian Woodford Shale Formation, Oklahoma, USA". In: *Interpretation* 6.1, SC55–SC66. ISSN: 2324-8858. DOI: 10.1190/INT-2017-0015.1. URL: https://doi.org/10.1190/INT-2017-0015.1 (cit. on p. 21).

Hale, D. (2013a). "Dynamic warping of seismic images". In: *GEOPHYSICS* 78.2, S105–S115. ISSN: 0016-8033. DOI: 10.1190/geo2012-0327.1. URL: http://library.seg.org/doi/10.1190/geo2012-0327.1 (cit. on pp. 5, 28).

Hale, D. (2013b). "Methods to compute fault images, extract fault surfaces, and estimate fault throws from 3D seismic images". In: *Geophysics* 78.2, O33–O43 (cit. on p. 22).

Hall, B. (2016). "Facies classification using machine learning". In: *Lead. Edge* 35.10, pp. 906–909. ISSN: 1070-485X. DOI: 10.1190/tle35100906.1. URL: https://doi.org/10.1190/tle35100906.1 (cit. on p. 21).

Hall, M. and B. Hall (2017). "Distributed collaborative prediction: Results of the machine learning contest". In: *Lead. Edge* 36.3, pp. 267–269. ISSN: 1070-485X. DOI: 10.1190/tle36030267.1. URL: https://doi.org/10.1190/tle36030267.1 (cit. on p. 21).

Hall, S. A., C. MacBeth, O. I. Barkved, and P. Wild (2002a). "Time-lapse seismic monitoring of compaction and subsidence at Valhall through cross-matching and interpreted warping of 3D streamer and OBC data". In: *SEG Technical Program Expanded Abstracts 2002*. Society of Exploration Geophysicists, pp. 1696–1699. DOI: 10.1190/1.1817004. URL: http://library.seg.org/doi/abs/10.1190/1.1817004 (cit. on p. 4).

Hall, S. A., C. MacBeth, O. I. Barkved, and P. Wild (2002b). "Time-lapse seismic monitoring of compaction and subsidence at Valhall through cross-matching and interpreted warping of 3D streamer and OBC data". In: *SEG Technical Program Expanded Abstracts 2002*. Society of Exploration Geophysicists, pp. 1696–1699 (cit. on p. 5).

Hansen, T. M. and K. S. Cordua (2017). "Efficient Monte Carlo sampling of inverse problems using a neural network-based forward—applied to GPR crosshole traveltime inversion". In: *Geophys. J. Int.* 211.3, pp. 1524–1533. ISSN: 0956-540X. DOI: 10.1093/gji/ggx380. URL: https://academic.oup.com/gji/article-abstract/211/3/1524/4157792 (cit. on p. 133).

Harrigan, E., J. R. Kroh, W. A. Sandham, and T. S. Durrani (1991). "Seismic wavelet extraction using artificial neural networks". In: *1991 Second International Conference on Artificial Neural Networks*. ieeexplore.ieee.org, pp. 95–99. URL: https://ieeexplore.ieee.org/abstract/document/140293/ (cit. on p. 133).

Hatchell, P. J., S. J. Bourne, and T. Netherlands. (2005a). "Measuring reservoir compaction using time-lapse timeshifts". In: *SEG/Houston 2005 Annual Meeting*, pp. 2500–2504 (cit. on p. 4).

Hatchell, P., S. Bourne, and T. Netherlands (2005b). "Rocks under strain: Strain-induced time-lapse time shifts are observed for depleting reservoirs". In: *Lead. Edge* 24.12, pp. 1222–1225. ISSN: 1070-485X. DOI: 10.1190/1.2149624. URL: http://library.seg.org/doi/10.1190/1.2149624 (cit. on p. 4).

He, K., X. Zhang, S. Ren, and J. Sun (2015). "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034 (cit. on p. 10).

He, K., X. Zhang, S. Ren, and J. Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778 (cit. on pp. 17, 131).

He, T., Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li (2019). "Bag of tricks for image classification with convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 558–567 (cit. on p. 131).

Helle, H. B. and A. Bhatt (2002). "Fluid saturation from well logs using committee neural networks". In: *Pet. Geosci.* ISSN: 1354-0793. URL: http://pg.lyellcollection.org/content/8/2/109.short (cit. on p. 134).

Herwanger, J. (2015). "Seismic Geomechanics - How to build and calibrate geomechanical models using 3D and 4D siesmic data". In: *EAGE Education Tour*, pp. 1–219 (cit. on p. 4).

Hinton, G. E. and R. R. Salakhutdinov (2006). "Reducing the dimensionality of data with neural networks". In: *science* 313.5786, pp. 504–507 (cit. on p. 18).

Hinton, G. E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov (2012). "Improving neural networks by preventing co-adaptation of feature detectors". In: *arXiv preprint arXiv:1207.0580* (cit. on p. 11).

Ho, T. K. (1995). "Random decision forests". In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE, pp. 278–282 (cit. on p. 8).

Hochreiter, S. (1991). "Untersuchungen zu dynamischen neuronalen Netzen". In: *Diploma, Technische Universität München* 91.1 (cit. on p. 10).

Hochreiter, S. and J. Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780 (cit. on p. 8).

Hopfield, J. J. (1982). "Neural networks and physical systems with emergent collective computational abilities". In: *Proceedings of the national academy of sciences* 79.8, pp. 2554–2558 (cit. on p. 8).

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Netw.* 2.5, pp. 359–366. ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90020-8. URL: http://www.sciencedirect.com/science/article/pii/0893608089900208 (cit. on p. 19).

Huang, G., Z. Liu, L. Van Der Maaten, and K. Q. Weinberger (2017a). "Densely connected convolutional networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708 (cit. on pp. 17, 131).

Huang, K. Y., W. R. I. Chang, and H. T. Yen (1990). "Self-organizing neural network for picking seismic horizons". In: *SEG Technical Program Expanded*. URL: https://library.seg.org/doi/pdf/10.1190/1.1890183 (cit. on p. 133).

Huang, L., X. Dong, and T. Clee (2017b). "A scalable deep learning platform for identifying geologic features from seismic attributes". In: *Lead. Edge* 36.3, pp. 249–256. ISSN: 1070-485X. DOI: 10.1190/tle36030249.1. URL: https://doi.org/10.1190/tle36030249.1 (cit. on p. 134).

Huang, Z., J. Shimeld, M. Williamson, and J. Katsube (1996). "Permeability prediction with artificial neural network modeling in the Venture gas field, offshore eastern Canada". In: *Geophysics* 61.2, pp. 422–436. ISSN: 0016-8033. DOI: 10.1190/1.1443970. URL: https://doi.org/10.1190/1.1443970 (cit. on p. 134).

Hulbert, C., B. Rouet-Leduc, C. X. Ren, J. Riviere, D. C. Bolton, C. Marone, and P. A. Johnson (2018). "Estimating the Physical State of a Laboratory Slow Slipping Fault from Seismic Signals". In: arXiv: 1801.07806 [physics.geo-ph]. URL: http://arxiv.org/abs/1801.07806 (cit. on p. 21).

Ioffe, S. and C. Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *arXiv preprint arXiv:1502.03167* (cit. on pp. 11, 131).

Iqbal, H. (2018). *HarisIqbal88/PlotNeuralNet v1.0.0*. DOI: 10.5281/zenodo.2526396. URL: https://doi.org/10.5281/zenodo.2526396.

Itakura, F. (1975). "Minimum Prediction Residual Principle Applied to Speech Recognition". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 23.1, pp. 67–72. ISSN: 00963518. DOI: 10.1109/TASSP.1975.1162641 (cit. on p. 29).

Iturrarán-Viveros, U. (2012). "Smooth regression to estimate effective porosity using seismic attributes". In: *J. Appl. Geophys.* 76, pp. 1–12. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2011.10.012. URL: http://www.sciencedirect.com/science/article/pii/S0926985111002485 (cit. on p. 134).

Iturrarán-Viveros, U. and J. O. Parra (2014). "Artificial Neural Networks applied to estimate permeability, porosity and intrinsic attenuation using seismic attributes and well-log data". In: *J. Appl. Geophys.* 107, pp. 45–54. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2014.05.010. URL: http://www.sciencedirect.com/science/article/pii/S092698511400144X (cit. on p. 133).

Jafrasteh, B., N. Fathianpour, and A. Suárez (2016). "Advanced Machine Learning Methods for Copper Ore Grade Estimation". In: *Near Surface Geoscience 2016*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=86600 (cit. on p. 133).

Jeong, J., E. Park, W. S. Han, and K.-Y. Kim (2014). "A novel data assimilation methodology for predicting lithology based on sequence labeling algorithms". In: *J. Geophys. Res. [Solid Earth]* 119.10, pp. 7503–7520. ISSN: 2169-9313. DOI: 10.1002/2014JB011279. URL: http://doi.wiley.com/10.1002/2014JB011279 (cit. on p. 22).

Johnston, D. H. (2013a). "6. Seismic Processing of 4D Data". In: *Practical Applications of Time-lapse Seismic Data*. Society of Exploration Geophysicists, pp. 103–126. DOI: 10.1190/1.9781560803126.ch6. URL: http://library.seg.org/doi/abs/10.1190/1.9781560803126.ch6 (cit. on pp. 3, 4).

Johnston, D. H. (2013b). *Practical Applications of Time-lapse Seismic Data*. Society of Exploration Geophysicists. ISBN: 9781560803072. DOI: 10.1190/1.9781560803126. URL: http://library.seg.org/doi/book/10.1190/1.9781560803126 (cit. on pp. 3, 4).

Kadurin, A., S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov (2017). "druGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico". en. In: *Mol. Pharm.* 14.9, pp. 3098–3104. ISSN: 1543-8384, 1543-8392. DOI: 10.1021/acs.molpharmaceut.7b00346. URL: http://dx.doi.org/10.1021/acs.molpharmaceut.7b00346 (cit. on p. 6).

Karra, S., D. O'Malley, J. D. Hyman, H. S. Viswanathan, and G. Srinivasan (2018). "Modeling flow and transport in fracture networks using graphs". en. In: *Phys Rev E* 97.3-1, p. 033304. ISSN: 2470-0053, 2470-0045. DOI: 10.1103/PhysRevE.97.033304. URL: http://dx.doi.org/10.1103/PhysRevE.97.033304 (cit. on p. 22).

Kelley, H. J. (1960). "Gradient theory of optimal flight paths". In: *Ars Journal* 30.10, pp. 947–954 (cit. on p. 8).

Keogh, E. and C. A. Ratanamahatana (2005). "Exact indexing of dynamic time warping". In: *Knowledge and information systems* 7.3, pp. 358–386 (cit. on pp. 29, 31).

Al-Khawari, H., R. P. Athyal, O. Al-Saeed, P. N. Sada, S. Al-Muthairi, and A. Al-Awadhi (2010). "Inter-and intraobserver variation between radiologists in the detection of abnormal parenchymal lung changes on high-resolution computed tomography". In: *Annals of Saudi medicine* 30.2, pp. 129–133 (cit. on p. 23).

Khoshnevis, N. and R. Taborda (2018). "Prioritizing ground-motion validation metrics using semisupervised and supervised learning". In: *Bull. Seismol. Soc. Am.* URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/108/4/2248/536309 (cit. on p. 22).

Kilic, G. and L. Eren (2018). "Neural network based inspection of voids and karst conduits in hydro–electric power station tunnels using GPR". In: *J. Appl. Geophys.* 151, pp. 194–204. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2018.02.026. URL: http://www.sciencedirect.com/science/article/pii/S0926985118301484 (cit. on p. 133).

Kingma, D. P. and J. Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv*. eprint: arXiv:1412.6980 (cit. on p. 11).

Kish, L. (1965). *Survey sampling*. 04; HN29, K5. (Cit. on p. 22).

Klose, C. D. (2006). "Self-organizing maps for geoscientific data analysis: geological interpretation of multidimensional geophysical data". In: *Computational Geosciences* 10.3, pp. 265–277. ISSN: 1573-1499. DOI: 10.1007/s10596-006-9022-x. URL: https://doi.org/10.1007/s10596-006-9022-x (cit. on p. 134).

Kohonen, T. (1982). "Self-organized formation of topologically correct feature maps". In: *Biological cybernetics* 43.1, pp. 59–69 (cit. on p. 13).

Kolmogorov, A. N. (1939). "Sur l'interpolation et extrapolation des suites stationnaires". In: *CR Acad Sci* 208, pp. 2043–2045 (cit. on p. 7).

Krige, D. G. (1951). "A statistical approach to some mine valuation and allied problems on the Witwatersrand". English. PhD thesis. Johannesburg (cit. on p. 7).

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105 (cit. on pp. 15, 131).

Krogh, A. and J. A. Hertz (1992). "A simple weight decay can improve generalization". In: *Advances in neural information processing systems*, pp. 950–957 (cit. on p. 11).

Kuehn, N. M., C. Riggelsen, et al. (2011). "Modeling the joint probability of earthquake, site, and ground-motion parameters using Bayesian networks". In: *Bulletin of the*. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/101/1/235/349494 (cit. on p. 22).

Kuroda, M. C., A. C. Vidal, and J. P. Papa (2016). "Analysis of porosity, stratigraphy, and structural delineation of a Brazilian carbonate field by machine learning techniques: A case study". In: *Interpretation* 4.3, T347–T358. ISSN: 2324-8858. DOI: 10.1190/INT-2016-0024.1. URL: https://pubs.geoscienceworld.org/interpretation/article-abstract/4/3/T347/309595 (cit. on p. 133).

Laake, A. (2014). "Structural interpretation in color—A new RGB processing application for seismic data". In: *Interpretation* 3.1, SC1–SC8 (cit. on p. 26).

Laloy, E., R. Hérault, D. Jacques, and N. Linde (2017). "Efficient training-image based geostatistical simulation and inversion using a spatial generative adversarial neural network". In: arXiv: 1708.04975 [stat.ML]. URL: http://arxiv.org/abs/1708.04975 (cit. on p. 21).

Landrø, M. (2001). "Discrimination between pressure and fluid saturation changes from time-lapse seismic data". In: *Geophysics* 66.3, pp. 836–844. ISSN: 0016-8033. DOI: 10.1190/1.1444973. URL: http://library.seg.org/doi/abs/10.1190/1.1444973 (cit. on p. 5).

Langer, H., G. Nunnari, and L. Occhipinti (1996). "Estimation of seismic waveform governing parameters with neural networks". In: *J. Geophys. Res.* 101.B9, pp. 20109–20118. ISSN: 0148-0227. DOI: 10.1029/96JB00948. URL: http://doi.wiley.com/10.1029/96JB00948 (cit. on p. 134).

Langer, H., S. Falsaperla, et al. (2003). "Application of artificial neural networks for the classification of the seismic transients at Soufriere Hills volcano, Montserrat". In: *Geophys. Res. Lett.* ISSN: 0094-8276. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2003GL018082 (cit. on p. 134).

Le Bouteiller, P., J. Charléty, F. Delprat-Jannaud, D. Granjeon, and C. Gorini (2018). "Mixing Unsupervised and Knowledge-Based Analysis for Heterogeneous Object Delineation in Seismic Data". In: *80th EAGE Conference and Exhibition 2018*. earthdoc.org. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92121 (cit. on p. 134).

LeCun, Y. A., L. Bottou, G. B. Orr, and K.-R. Müller (2012). "Efficient backprop". In: *Neural networks: Tricks of the trade*. Springer, pp. 9–48 (cit. on p. 10).

LeCun, Y., Y. Bengio, and G. Hinton (2015). "Deep learning". In: *nature* 521.7553, pp. 436–444 (cit. on p. 8).

Legget, M., W. A. Sandham, and T. S. Durrani (1996). "3D horizon tracking using artificial neural networks". In: *First Break*. ISSN: 0263-5046. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=28049 (cit. on pp. 20, 133).

Leggett, M., W. A. Sandham, and T. S. Durrani (2003). "Automated 3-D Horizon Tracking and Seismic Classification Using Artificial Neural Networks". In: *Geophysical Applications of Artificial Neural Networks and Fuzzy Logic*. Ed. by W. A. Sandham and M. Leggett. Dordrecht: Springer Netherlands, pp. 31–44. ISBN: 9789401702713. DOI: 10.1007/978-94-017-0271-3_3. URL: https://doi.org/10.1007/978-94-017-0271-3_3 (cit. on pp. 21, 133).

Lewis, W. and D. Vigh (2017). "Deep learning prior models from seismic images for full-waveform inversion". In: *SEG Technical Program Expanded Abstracts 2017*. SEG Technical Program Expanded Abstracts. Society of Exploration Geophysicists, pp. 1512–1517. DOI: 10.1190/segam2017-17627643.1. URL: https://doi.org/10.1190/segam2017-17627643.1 (cit. on p. 134).

Li, J. and J. Castagna (2004). "Support Vector Machine (SVM) pattern recognition to AVO classification". In: *Geophys. Res. Lett.* 31.2, p. 948. ISSN: 0094-8276. DOI:

10.1029/2003GL018299. URL: http://doi.wiley.com/10.1029/2003GL018299 (cit. on pp. 20, 21).

Li, L., X. W. Li, Z. Wan, Y. Liu, and L. Zhang (2018). "Multiscale Pre-Stack Seismic Attribute Enhancement Using Radial Basis Function Network". In: *80th EAGE Conference and*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92122 (cit. on p. 134).

Linnainmaa, S. (1970). "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors". In: *Master's Thesis (in Finnish), Univ. Helsinki*, pp. 6–7 (cit. on p. 8).

Liu, X., P. He, W. Chen, and J. Gao (2019). "Multi-task deep neural networks for natural language understanding". In: *arXiv preprint arXiv:1901.11504* (cit. on p. 24).

Liu, Y., Z. Chen, L. Wang, Y. Zhang, Z. Liu, and Y. Shuai (2015). "Quantitative seismic interpretations to detect biogenic gas accumulations: a case study from Qaidam Basin, China". In: *Bull. Can. Petrol. Geol.* 63.1, pp. 108–121. ISSN: 0007-4802. DOI: 10.2113/gscpgbull.63.1.108. URL: https://pubs.geoscienceworld.org/cspg/bcpg/article-abstract/63/1/108/455952 (cit. on p. 21).

Londoño, J. M. and H. Kumagai (2018). "4D seismic tomography of Nevado del Ruiz Volcano, Colombia, 2000–2016". In: *Journal of Volcanology and Geothermal Research* 358, pp. 105–123 (cit. on p. 3).

Lumley, D. E. (1995). *Seismic time-lapse monitoring of subsurface fluid flow*. 91. Stanford University (cit. on p. 3).

Lumley, D. E. (2001). "Time-lapse seismic reservoir monitoring". In: *Geophysics* 66.1, pp. 50–53. ISSN: 0016-8033. DOI: 10.1190/1.1444921. URL: http://library.seg.org/doi/abs/10.1190/1.1444921 (cit. on p. 4).

Lundberg, S. M. and S.-I. Lee (2017). "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., pp. 4765–4774. URL: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf (cit. on p. 24).

Luo, S. and D. Hale (2014). "Least-squares migration in the presence of velocity errors". In: *SEG Technical Program Expanded Abstracts 2014*. 5. Society of Exploration Geophysicists, pp. 3980–3984. DOI: 10.1190/segam2014-1367.1. URL: http://library.seg.org/doi/abs/10.1190/segam2014-1367.1 (cit. on p. 28).

Ma, J., Z. Jiang, Q. Tian, and G. D. Couples (2012). "Classification of Digital Rocks by Machine Learning". In: *ECMOR XIII-13th European*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=62262 (cit. on p. 21).

Maas, C. and J. Schmalzl (2013). "Using pattern recognition to automatically localize reflection hyperbolas in data from ground penetrating radar". In: *Comput. Geosci.* 58, pp. 116–125. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2013.04.012. URL: http://www.sciencedirect.com/science/article/pii/S009830041300112X (cit. on p. 133).

MacBeth, C., M.-D. Mangriotis, and H. Amini (2019). "Post-stack 4D seismic time-shifts: Interpretation and evaluation". In: *Geophysical Prospecting* 67.1, pp. 3–31 (cit. on p. 4).

Maggi, A., V. Ferrazzini, C. Hibert, F. Beauducel, P. Boissier, and A. Amemoutou (2017). "Implementation of a Multistation Approach for Automated Event Classification at Piton de la Fournaise Volcano". In: *Seismol. Res. Lett.* 88.3, pp. 878–891. ISSN: 0895-0695. DOI: 10.1785/0220160189. URL: https://pubs.geoscienceworld.org/ssa/srl/article-abstract/88/3/878/284054 (cit. on p. 21).

Maiti, S. and R. K. Tiwari (2010). "Neural network modeling and an uncertainty analysis in Bayesian framework: A case study from the KTB borehole site". In: *J. Geophys. Res.* 115.B10, E67. ISSN: 0148-0227. DOI: 10.1029/2010JB000864. URL: http://doi.wiley.com/10.1029/2010JB000864 (cit. on p. 134).

Malfante, M., M. D. Mura, J. Metaxian, J. I. Mars, O. Macedo, and A. Inza (2018). "Machine Learning for Volcano-Seismic Signals: Challenges and Perspectives". In: *IEEE Signal Process. Mag.* 35.2, pp. 20–30. ISSN: 1053-5888. DOI: 10.1109/MSP.2017.2779166. URL: http://dx.doi.org/10.1109/MSP.2017.2779166 (cit. on p. 21).

Mardan, A., A. Javaherian, et al. (2017). "Channel Characterization Using Support Vector Machine". In: *79th EAGE Conference*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=89283 (cit. on p. 21).

Marjanović, M., M. Kovačević, B. Bajat, and V. Voženílek (2011). "Landslide susceptibility assessment using SVM machine learning algorithm". In: *Eng. Geol.* 123.3, pp. 225–234. ISSN: 0013-7952. DOI: 10.1016/j.enggeo.2011.09.006. URL: http://www.sciencedirect.com/science/article/pii/S0013795211002195 (cit. on p. 21).

Markov, A. A. (1906). "Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga". In: *Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete* 15.135-156, p. 18 (cit. on pp. 7, 171).

Markov, A. A. (1971). "Extension of the limit theorems of probability theory to a sum of variables connected in a chain". In: *Dynamic probabilistic systems* 1. Reprint in English of (Markov, 1906), pp. 552–577 (cit. on p. 7).

Marroquín, I. (2014). "A knowledge-integration framework for interpreting seismic facies". In: *Interpretation* 2.1, SA1–SA9. ISSN: 2324-8858. DOI: 10.1190/INT-2013-0057.1. URL: https://doi.org/10.1190/INT-2013-0057.1 (cit. on p. 134).

Martinelli, G., J. Eidsvik, R. Sinding-Larsen, et al. (2013). "Building Bayesian networks from basin-modelling scenarios for improved geological decision making". In: *Petroleum*. URL: http://pg.lyellcollection.org/content/early/2013/06/24/petgeo2012-057.abstract (cit. on p. 22).

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Y. Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng (2015). *TensorFlow: Large-Scale Ma-*

chine Learning on Heterogeneous Systems. Software available from tensorflow.org. URL: http://tensorflow.org/ (cit. on pp. xi, 6, 8).

Masotti, M., R. Campanini, L. Mazzacurati, S. Falsaperla, H. Langer, and S. Spampinato (2008). "TREMOrEC: A software utility for automatic classification of volcanic tremor". In: *Geochem. Geophys. Geosyst.* 9.4. ISSN: 1525-2027. DOI: 10.1029/2007GC001860. URL: http://doi.wiley.com/10.1029/2007GC001860 (cit. on p. 21).

Masotti, M., S. Falsaperla, H. Langer, S. Spampinato, and R. Campanini (2006). "Application of Support Vector Machine to the classification of volcanic tremor at Etna, Italy". In: *Geophys. Res. Lett.* 33.20, p. 113. ISSN: 0094-8276. DOI: 10.1029/2006GL027441. URL: http://doi.wiley.com/10.1029/2006GL027441 (cit. on p. 21).

McCormack, M. (1991). "Neural computing in geophysics". In: *Lead. Edge* 10.1, pp. 11–15. ISSN: 1070-485X. DOI: 10.1190/1.1436771. URL: https://doi.org/10.1190/1.1436771 (cit. on p. 19).

McCormack, M. D., D. E. Zaucha, and D. W. Dushek (1993). "First-break refraction event picking and seismic data trace editing using neural networks". In: *Geophysics*. ISSN: 0016-8033. URL: https://library.seg.org/doi/abs/10.1190/1.1443352 (cit. on pp. 133, 134).

McErlean, A., D. M. Panicek, E. C. Zabor, C. S. Moskowitz, R. Bitar, R. J. Motzer, H. Hricak, and M. S. Ginsberg (2013). "Intra- and Interobserver Variability in CT Measurements in Oncology". In: *Radiology* 269.2, pp. 451–459. DOI: 10.1148/radiol.13122665. URL: https://doi.org/10.1148/radiol.13122665 (cit. on p. 23).

McKinney, W. et al. (2010). "Data structures for statistical computing in python". In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX, pp. 51–56 (cit. on p. xi).

Meier, U., A. Curtis, and J. Trampert (2007a). "Fully nonlinear inversion of fundamental mode surface waves for a global crustal model". In: *Geophys. Res. Lett.* 34.16, p. 151. ISSN: 0094-8276. DOI: 10.1029/2007GL030989. URL: http://doi.wiley.com/10.1029/2007GL030989 (cit. on p. 134).

Meier, U., A. Curtis, and J. Trampert (2007b). "Global crustal thickness from neural network inversion of surface wave data". In: *Geophys. J. Int.* 169.2, pp. 706–722. ISSN: 0956-540X. DOI: 10.1111/j.1365-246X.2007.03373.x. URL: https://academic.oup.com/gji/article-abstract/169/2/706/2012014 (cit. on p. 134).

Meldahl, P., R. Heggland, B. Bril, and P. de Groot (2001). "Identifying faults and gas chimneys using multiattributes and neural networks". In: *Lead. Edge* 20.5, pp. 474–482. ISSN: 1070-485X. DOI: 10.1190/1.1438976. URL: https://doi.org/10.1190/1.1438976 (cit. on p. 134).

Mertens, L., R. Persico, L. Matera, et al. (2016). "Automated Detection of Reflection Hyperbolas in Complex GPR Images With No A Priori Knowledge on the Medium". In: *IEEE Transactions on*. URL: https://ieeexplore.ieee.org/abstract/document/7230274/ (cit. on p. 133).

Mignan, A. and M. Broccardo (2019a). "A Deeper Look into 'Deep Learning of Aftershock Patterns Following Large Earthquakes': Illustrating First Principles in Neural

Network Physical Interpretability". In: *International Work-Conference on Artificial Neural Networks*. Springer, pp. 3–14 (cit. on p. 21).

Mignan, A. and M. Broccardo (2019b). "One neuron versus deep learning in aftershock prediction". In: *Nature* 574.7776, E1–E3 (cit. on p. 21).

Mjolsness, E. and D. DeCoste (2001). "Machine learning for science: state of the art and future prospects". en. In: *Science* 293.5537, pp. 2051–2055. ISSN: 0036-8075. DOI: 10.1126/science.293.5537.2051. URL: http://dx.doi.org/10.1126/science.293.5537.2051 (cit. on p. 20).

Mosser, L., O. Dubrule, and M. J. Blunt (2017). "Reconstruction of three-dimensional porous media using generative adversarial neural networks". en. In: *Phys Rev E* 96.4-1, p. 043309. ISSN: 2470-0053, 2470-0045. DOI: 10.1103/PhysRevE.96.043309. URL: http://dx.doi.org/10.1103/PhysRevE.96.043309 (cit. on pp. 21, 133).

Mosser, L., O. Dubrule, and M. J. Blunt (2018a). "Conditioning of three-dimensional generative adversarial networks for pore and reservoir-scale models". In: arXiv: 1802.05622 [stat.ML]. URL: http://arxiv.org/abs/1802.05622 (cit. on p. 133).

Mosser, L., O. Dubrule, and M. J. Blunt (2018b). "Stochastic Reconstruction of an Oolitic Limestone by Generative Adversarial Networks". In: *Transp. Porous Media* 125.1, pp. 81–103. ISSN: 1573-1634. DOI: 10.1007/s11242-018-1039-9. URL: https://doi.org/10.1007/s11242-018-1039-9 (cit. on p. 133).

Mosser, L., O. Dubrule, and M. J. Blunt (2018c). "Stochastic seismic waveform inversion using generative adversarial networks as a geological prior". In: arXiv: 1806.03720 [physics.geo-ph]. URL: http://arxiv.org/abs/1806.03720 (cit. on pp. 21, 134).

Mosser, L., W. Kimman, J. Dramsch, S. Purves, et al. (2018d). "Rapid seismic domain transfer: Seismic velocity inversion and modeling using deep generative neural networks". In: *EAGE Conference and ….* URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92120 (cit. on pp. 21, 134).

Murat, M. E. and A. J. Rudman (1992). "AUTOMATED FIRST ARRIVAL PICKING: A NEURAL NETWORK APPROACH1". In: *Geophys. Prospect.* 40.6, pp. 587–604. ISSN: 0016-8025, 1365-2478. DOI: 10.1111/j.1365-2478.1992.tb00543.x. URL: http://doi.wiley.com/10.1111/j.1365-2478.1992.tb00543.x (cit. on p. 133).

Musil, M. and A. Plešinger (1996a). "Discrimination between local microearthquakes and quarry blasts by multi-layer perceptrons and Kohonen maps". In: *Bull. Seismol. Soc. Am.* 86.4, pp. 1077–1090. ISSN: 0037-1106. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/86/4/1077/120135 (cit. on p. 134).

Musil, M. and A. Plešinger (1996b). "Discrimination between local microearthquakes and quarry blasts by multi-layer perceptrons and Kohonen maps". In: *Bull. Seismol. Soc. Am.* 86.4, pp. 1077–1090. ISSN: 0037-1106. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/86/4/1077/120135 (cit. on p. 134).

Nash, J. (1951). "Non-cooperative games". In: *Annals of mathematics*, pp. 286–295 (cit. on p. 16).

Newendorp, P. D. (1976). "Decision analysis for petroleum exploration". In: (cit. on p. 20).

Al-Nuaimy, W., Y. Huang, M. Nakhkash, M. T. C. Fang, V. T. Nguyen, and A. Eriksen (2000). "Automatic detection of buried utilities and solid objects with GPR using

neural networks and pattern recognition". In: *J. Appl. Geophys.* 43.2, pp. 157–165. ISSN: 0926-9851. DOI: 10.1016/S0926-9851(99)00055-5. URL: http://www.sciencedirect.com/science/article/pii/S0926985199000555 (cit. on p. 133).

Núñez-Nieto, X., M. Solla, P. Gómez-Pérez, and H. Lorenzo (2014). "GPR Signal Characterization for Automated Landmine and UXO Detection Based on Machine Learning Techniques". en. In: *Remote Sensing* 6.10, pp. 9729–9748. DOI: 10.3390/rs6109729. URL: https://www.mdpi.com/2072-4292/6/10/9729/htm (cit. on p. 133).

Ochoa, L. H., L. F. Niño, and C. A. Vargas (2018). "Fast magnitude determination using a single seismological station record implementing machine learning techniques". In: *Geodesy and Geodynamics* 9.1, pp. 34–41. ISSN: 1674-9847. DOI: 10.1016/j.geog.2017.03.010. URL: http://www.sciencedirect.com/science/article/pii/S1674984717300058 (cit. on p. 21).

Oh, H.-J. and S. Lee (2010). "Application of Artificial Neural Network for Gold–Silver Deposits Potential Mapping: A Case Study of Korea". In: *Nat. Resour. Res.* 19.2, pp. 103–124. ISSN: 1520-7439, 1573-8981. DOI: 10.1007/s11053-010-9112-2. URL: https://doi.org/10.1007/s11053-010-9112-2 (cit. on p. 133).

Ohrnberger, M. (2001). *[No title]*. https://www.researchgate.net/profile/Matthias_Ohrnberger/publication/252958874_Continuous_Automatic_Classification_of_Seismic_Signals_of_Volcanic_Origin_at_Mt_Merapi_Java_Indonesia/links/573ffe3e08aea45ee84504a4/Continuous-Automatic-Classification-of-Seismic-Signals-of-Volcanic-Origin-at-Mt-Merapi-Java-Indonesia.pdf. Accessed: 2018-12-17. URL: https://www.researchgate.net/profile/Matthias_Ohrnberger/publication/252958874_Continuous_Automatic_Classification_of_Seismic_Signals_of_Volcanic_Origin_at_Mt_Merapi_Java_Indonesia/links/573ffe3e08aea45ee84504a4/Continuous-Automatic-Classification-of-Seismic-Signals-of-Volcanic-Origin-at-Mt-Merapi-Java-Indonesia.pdf (cit. on p. 22).

Pasolli, E., F. Melgani, and M. Donelli (2009). "Automatic analysis of GPR images: A pattern-recognition approach". In: *IEEE Transactions on Geoscience and Remote Sensing* 47.7, pp. 2206–2217 (cit. on p. 21).

Paszke, A., S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer (2017). "Automatic Differentiation in PyTorch". In: *NIPS Autodiff Workshop* (cit. on p. 6).

Patel, A. K. and S. Chatterjee (2016). "Computer vision-based limestone rock-type classification using probabilistic neural network". In: *Geoscience Frontiers* 7.1, pp. 53–60. ISSN: 1674-9871. DOI: 10.1016/j.gsf.2014.10.005. URL: http://www.sciencedirect.com/science/article/pii/S1674987114001388 (cit. on p. 134).

Pedersen, S. I., T. Randen, L. Sonneland, et al. (2002). "Automatic fault extraction using artificial ants". In: *SEG Technical Program*. URL: https://library.seg.org/doi/pdf/10.1190/1.1817297 (cit. on p. 22).

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning

in Python". In: *Journal of Machine Learning Research* 12, pp. 2825–2830 (cit. on pp. xii, 6, 8).

Porwal, A., E. J. M. Carranza, and M. Hale (2003). "Artificial Neural Networks for Mineral-Potential Mapping: A Case Study from Aravalli Province, Western India". In: *Nat. Resour. Res.* 12.3, pp. 155–171. ISSN: 1520-7439, 1573-8981. DOI: 10.1023/A:1025171803637. URL: https://doi.org/10.1023/A:1025171803637 (cit. on p. 133).

Poulton, M., B. Sternberg, and C. Glass (1992). "Location of subsurface targets in geophysical data using neural networks". In: *Geophysics* 57.12, pp. 1534–1544. ISSN: 0016-8033. DOI: 10.1190/1.1443221. URL: https://doi.org/10.1190/1.1443221 (cit. on p. 20).

Preston, F. W. and J. Henderson (1964). *Fourier series characterization of cyclic sediments for stratigraphic correlation*. Kansas Geological Survey (cit. on p. 20).

Purves, S., B. Alaei, and E. Larsen (2018). "Bootstrapping Machine-Learning Based Seismic Fault Interpretation". In: *ACE 2018 Annual Convention &*. URL: http://www.searchanddiscovery.com/abstracts/html/2018/ace2018/abstracts/2856016.html (cit. on p. 134).

Purves, S., B. Alaei, and D. Lolis (2019). "Towards Subsurface ML Metrics". In: *81st EAGE Conference and Exhibition 2019 Workshop Programme* (cit. on p. 23).

Qi, J., T. Lin, T. Zhao, F. Li, and K. Marfurt (2016). "Semisupervised multiattribute seismic facies analysis". In: *Interpretation* 4.1, SB91–SB106. ISSN: 2324-8858. DOI: 10.1190/INT-2015-0098.1. URL: https://doi.org/10.1190/INT-2015-0098.1 (cit. on p. 134).

Real, E., A. Aggarwal, Y. Huang, and Q. V. Le (2019). "Regularized evolution for image classifier architecture search". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33, pp. 4780–4789 (cit. on pp. 17, 131).

Reddy, R. and G. Bonham-Carter (1991). "A Decision-Tree Approach to Mineral Potential Mapping in Snow Lake Area, Manitoba". In: *Canadian Journal of Remote Sensing* 17.2, pp. 191–200. DOI: 10.1080/07038992.1991.10855292. eprint: https://doi.org/10.1080/07038992.1991.10855292. URL: https://doi.org/10.1080/07038992.1991.10855292 (cit. on p. 20).

Ribeiro, M. T., S. Singh, and C. Guestrin (2016). "Why should i trust you?: Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp. 1135–1144 (cit. on p. 24).

Richardson, A. (2018). "Seismic Full-Waveform Inversion Using Deep Learning Tools and Techniques". In: arXiv: 1801.07232 [physics.geo-ph]. URL: http://arxiv.org/abs/1801.07232 (cit. on p. 134).

Rickett, J. E. and D. Lumley (2001). "Cross-equalization data processing for time-lapse seismic reservoir monitoring: A case study from the Gulf of Mexico". In: *Geophysics* 66.4, pp. 1015–1025. ISSN: 0016-8033. DOI: 10.1190/1.1487049. URL: http://library.seg.org/doi/abs/10.1190/1.1487049 (cit. on p. 5).

Rickett, J., L. Duranti, S. Ramon, U. T. Hudson, B. Regel, and N. Orleans (2007). "4D time strain and the seismic signature of geomechanical compaction at Genesis". In:

Lead. Edge 26.5, pp. 644–647. ISSN: 1070-485X. DOI: 10.1190/1.2737103. URL: http://library.seg.org/doi/10.1190/1.2737103 (cit. on p. 5).

Roden, R., T. Smith, and D. Sacrey (2015). "Geologic pattern recognition from seismic attributes: Principal component analysis and self-organizing maps". In: *Interpretation* 3.4, SAE59–SAE83. ISSN: 2324-8858. DOI: 10.1190/INT-2015-0037.1. URL: https://pubs.geoscienceworld.org/interpretation/article-abstract/3/4/SAE59/75892 (cit. on p. 134).

Romeo, G. (1994). "Seismic signals detection and classification using artiricial neural networks". In: *Annals of Geophysics* 37.3. ISSN: 2037-416X, 2037-416X. DOI: 10.4401/ag-4211. URL: http://www.annalsofgeophysics.eu/index.php/annals/article/view/4211 (cit. on p. 134).

Ronneberger, O., P. Fischer, and T. Brox (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241 (cit. on pp. 18, 19).

Rosenblatt, F. (1958). "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6, p. 386 (cit. on p. 7).

Ross, Z. E., M. A. Meier, and E. Hauksson (2018a). "P-wave arrival picking and first-motion polarity determination with deep learning". In: *J. Geophys. Res.* ISSN: 0148-0227. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2017JB015251 (cit. on pp. 21, 133).

Ross, Z. E., M.-A. Meier, E. Hauksson, and T. H. Heaton (2018b). "Generalized Seismic Phase Detection with Deep Learning". In: arXiv: 1805.01075 [physics.geo-ph]. URL: http://arxiv.org/abs/1805.01075 (cit. on pp. 21, 134).

Røste, T., A. Stovas, and M. Landrø (2006). "Estimation of layer thickness and velocity changes using 4D prestack seismic data". In: *Geophysics* 71.6, S219–S234 (cit. on p. 4).

Röth, G. and A. Tarantola (1994). "Neural networks and inversion of seismic data". In: *J. Geophys. Res.* 99.B4, p. 6753. ISSN: 0148-0227. DOI: 10.1029/93JB01563. URL: http://doi.wiley.com/10.1029/93JB01563 (cit. on pp. 20, 134).

Rouet-Leduc, B., C. Hulbert, N. Lubbers, et al. (2017). "Machine learning predicts laboratory earthquakes". In: *Geophysical*. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2017GL074677 (cit. on p. 21).

Rouet-Leduc, B., C. Hulbert, D. C. Bolton, et al. (2018). "Estimating fault friction from seismic signals in the laboratory". In: *Geophysical*. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2017GL076708 (cit. on p. 21).

Rumelhart, D. E., G. E. Hinton, R. J. Williams, et al. (1988). "Learning representations by back-propagating errors". In: *Cognitive modeling* 5.3, p. 1 (cit. on p. 8).

Russakovsky, O., J. Deng, Z. Huang, A. C. Berg, and L. Fei-Fei (2013). "Detecting avocados to zucchinis: what have we done, and where are we going?" In: *International Conference on Computer Vision (ICCV)* (cit. on p. 8).

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei (2015). "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252. DOI: 10.1007/s11263-015-0816-y (cit. on p. 15).

Russell, S. J. and P. Norvig (2010). *Artificial Intelligence - A Modern Approach, Third International Edition*. Pearson Education. ISBN: 978-0-13-207148-2. URL: http://vig.pearsoned.com/store/product/1,1207,store-12521%5C_isbn-0136042597,00.html (cit. on p. 7).

Sacrey, D. and R. Roden (2018). "Solving exploration problems with machine learning". In: *First Break*. ISSN: 0263-5046. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92017 (cit. on p. 134).

Sahoo, S., T. A. Russo, J. Elliott, et al. (2017). "Machine learning algorithms for modeling groundwater level changes in agricultural regions of the US". In: *Water Resour.* URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2016WR019933 (cit. on p. 21).

Sakoe, H. and S. Chiba (1978). "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1, pp. 43–49. ISSN: 0096-3518. DOI: 10.1109/TASSP.1978.1163055. URL: http://ieeexplore.ieee.org/document/1163055/ (cit. on p. 29).

Saporetti, C. M., L. G. da Fonseca, E. Pereira, and L. C. de Oliveira (2018). "Machine learning approaches for petrographic classification of carbonate-siliciclastic rocks using well logs and textural information". In: *J. Appl. Geophys.* 155, pp. 217–225. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2018.06.012. URL: http://www.sciencedirect.com/science/article/pii/S092698511630667X (cit. on pp. 21, 22, 134).

Scarpetta, S., F. Giudicepietro, E. C. Ezin, S. Petrosino, E. Del Pezzo, M. Martini, and M. Marinaro (2005). "Automatic Classification of Seismic Signals at Mt. Vesuvius Volcano, Italy, Using Neural Networks". In: *Bull. Seismol. Soc. Am.* 95.1, pp. 185–196. ISSN: 0037-1106. DOI: 10.1785/0120030075. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/95/1/185/146889 (cit. on p. 134).

Schuster, G. T. S. (2018). "Machine learning and wave equation inversion of skeletonized data". In: *80th EAGE Conference & Exhibition 2018 Workshop*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=93370 (cit. on p. 134).

Schütt, K. T., F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko (2017). "Quantum-chemical insights from deep tensor neural networks". en. In: *Nat. Commun.* 8, p. 13890. ISSN: 2041-1723. DOI: 10.1038/ncomms13890. URL: http://dx.doi.org/10.1038/ncomms13890 (cit. on p. 6).

Schwarzacher, W. (1972). "The semi-Markov process as a general sedimentation model". In: *Mathematical Models of Sedimentary Processes*. Springer, pp. 247–268 (cit. on pp. 20, 22).

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra (2017). "Grad-cam: Visual explanations from deep networks via gradient-based localization". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626 (cit. on p. 24).

Shafiq, M. A., M. Prabhushankar, and G. AlRegib (2018a). "Leveraging Sparse Features Learned from Natural Images for Seismic Understanding". In: *80th EAGE Confer-*

ence and. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92123 (cit. on p. 134).

Shafiq, M. A., M. Prabhushankar, and G. AlRegib (2018b). "Leveraging Sparse Features Learned from Natural Images for Seismic Understanding". In: *80th EAGE Conference and*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92123 (cit. on p. 134).

Shafiq, M. A., M. Prabhushankar, et al. (2018c). "Attention models based on sparse autoencoders for seismic interpretation". In: *ACE 2018 Annual*. URL: http://www.searchanddiscovery.com/abstracts/html/2018/ace2018/abstracts/2856356.html (cit. on p. 134).

Shen, D., G. Wu, and H.-I. Suk (2017). "Deep Learning in Medical Image Analysis". en. In: *Annu. Rev. Biomed. Eng.* 19, pp. 221–248. ISSN: 1523-9829, 1545-4274. DOI: 10.1146/annurev-bioeng-071516-044442. URL: http://dx.doi.org/10.1146/annurev-bioeng-071516-044442 (cit. on p. 6).

Shihab, S., W. Al-Nuaimy, and A. Eriksen (2002a). "Image processing and neural network techniques for automatic detection and interpretation of ground penetrating radar data". In: *Proceedings of the 6th*. URL: https://www.researchgate.net/profile/Asger_Eriksen/publication/255015233_Image_processing_and_neural_network_techniques_for_automatic_detection_and_interpretation_of_ground_penetrating_radar_data/links/54c664b60cf219bbe4f842ba/Image-processing-and-neural-network-techniques-for-automatic-detection-and-interpretation-of-ground-penetrating-radar-data.pdf (cit. on p. 133).

Shihab, S., W. Al-Nuaimy, Y. Huang, and A. Eriksen (2002b). "Neural network target identifier based on statistical features of GPR signals". In: *Ninth International Conference on Ground Penetrating Radar*. Vol. 4758. International Society for Optics and Photonics, pp. 135–139. DOI: 10.1117/12.462228. URL: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/4758/0000/Neural-network-target-identifier-based-on-statistical-features-of-GPR/10.1117/12.462228.short (cit. on p. 133).

Shrikumar, A., P. Greenside, and A. Kundaje (2017). "Learning important features through propagating activation differences". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 3145–3153 (cit. on p. 24).

Simonyan, K. and A. Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (cit. on p. 131).

Strecker, U. and R. Uden (2002). "Data mining of 3D poststack seismic attribute volumes using Kohonen self-organizing maps". In: *Lead. Edge* 21.10, pp. 1032–1037. ISSN: 1070-485X. DOI: 10.1190/1.1518442. URL: https://doi.org/10.1190/1.1518442 (cit. on p. 134).

Su, J., D. V. Vargas, and K. Sakurai (2019). "One pixel attack for fooling deep neural networks". In: *IEEE Transactions on Evolutionary Computation* (cit. on p. 23).

Sudakov, O., E. Burnaev, and D. Koroteev (2018). "Driving Digital Rock towards Machine Learning: predicting permeability with Gradient Boosting and Deep Neural

Networks". In: arXiv: 1803.00758 [physics.geo-ph]. URL: http://arxiv.org/abs/1803.00758 (cit. on p. 133).

Sutskever, I., J. Martens, G. Dahl, and G. Hinton (2013). "On the importance of initialization and momentum in deep learning". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by S. Dasgupta and D. McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, pp. 1139–1147. URL: http://proceedings.mlr.press/v28/sutskever13.html (cit. on p. 11).

Szegedy, C., S. Ioffe, V. Vanhoucke, and A. A. Alemi (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-First AAAI Conference on Artificial Intelligence* (cit. on p. 131).

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich (2015). "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9 (cit. on pp. 17, 131).

Szegedy, C., V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826 (cit. on p. 131).

Tan, M., B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le (2019a). "Mnasnet: Platform-aware neural architecture search for mobile". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2820–2828 (cit. on p. 131).

Tan, M. and Q. V. Le (2019b). "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: *arXiv preprint arXiv:1905.11946* (cit. on pp. 17, 131).

Tan, M. and Q. V. Le (2019c). "MixConv: Mixed Depthwise Convolutional Kernels". In: *CoRR, abs/1907.09595* (cit. on p. 131).

Tartakovsky, D. M. (2004). "Delineation of geologic facies with statistical learning theory". In: *Geophys. Res. Lett.* 31.18, p. 121. ISSN: 0094-8276. DOI: 10.1029/2004GL020864. URL: http://doi.wiley.com/10.1029/2004GL020864 (cit. on p. 20).

Taylor, R. (1843). *Scientific memoirs, selected from the transactions of foreign academies of science and learned societies, and from foreign journals*. Vol. 3. R. and JE Taylor (cit. on p. 7).

Terzaghi, R. D. (1965). "Sources of error in joint surveys". In: *Geotechnique* 15.3, pp. 287–304 (cit. on p. 27).

Touvron, H., A. Vedaldi, M. Douze, and H. Jégou (2019). "Fixing the train-test resolution discrepancy". In: *arXiv preprint arXiv:1906.06423* (cit. on p. 131).

Trabelsi, C., O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal (2017). "Deep complex networks". In: *arXiv preprint arXiv:1705.09792* (cit. on pp. 38, 135).

Turing, A. M. (1950). "I.—Computing Machinery and Intelligence". In: *Mind* LIX.236, pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. eprint: http://oup.prod.sis.lan/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf. URL: https://doi.org/10.1093/mind/LIX.236.433 (cit. on p. 7).

Valera, M., Z. Guo, P. Kelly, S. Matz, V. A. Cantu, A. G. Percus, J. D. Hyman, G. Srinivasan, and H. S. Viswanathan (2017). "Machine learning for graph-based representations of three-dimensional discrete fracture networks". In: arXiv: 1705.09866 [physics.geo-ph]. URL: http://arxiv.org/abs/1705.09866 (cit. on p. 21).

Veillard, A., O. Morère, M. Grout, et al. (2018). "Fast 3D Seismic Interpretation with Unsupervised Deep Learning: Application to a Potash Network in the North Sea". In: *80th EAGE Conference*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92124 (cit. on p. 134).

Verma, A. K., S. Chaki, A. Routray, W. K. Mohanty, and M. Jenamani (2014). "Quantification of sand fraction from seismic attributes using Neuro-Fuzzy approach". In: *J. Appl. Geophys.* 111, pp. 141–155. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2014.10.005. URL: http://www.sciencedirect.com/science/article/pii/S0926985114002912 (cit. on p. 134).

Waldeland, A. U. and A. Solberg (2017). "Salt classification using deep learning". In: *79th EAGE Conference and Exhibition*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=88635 (cit. on p. 134).

Waldeland, A., A. Jensen, L. Gelius, and A. Solberg (2018). "Convolutional neural networks for automated seismic interpretation". In: *Lead. Edge* 37.7, pp. 529–537. ISSN: 1070-485X. DOI: 10.1190/tle37070529.1. URL: https://doi.org/10.1190/tle37070529.1 (cit. on p. 134).

Wang, H., J. F. Wellmann, Z. Li, X. Wang, and R. Y. Liang (2017a). "A Segmentation Approach for Stochastic Geological Modeling Using Hidden Markov Random Fields". In: *Math. Geosci.* 49.2, pp. 145–177. ISSN: 1874-8961, 1874-8953. DOI: 10.1007/s11004-016-9663-9. URL: https://doi.org/10.1007/s11004-016-9663-9 (cit. on p. 22).

Wang, J. and T.-L. Teng (1997). "Identification and picking of S phase using an artificial neural network". In: *Bull. Seismol. Soc. Am.* 87.5, pp. 1140–1149. ISSN: 0037-1106. URL: https://pubs.geoscienceworld.org/ssa/bssa/article-abstract/87/5/1140/120211 (cit. on pp. 133, 134).

Wang, K., J. Lomask, and F. Segovia (2017b). "Automatic, geologic layer-constrained well-seismic tie through blocked dynamic warping". In: *Interpretation* 5.3, SJ81–SJ90. ISSN: 2324-8858. DOI: 10.1190/INT-2016-0160.1. URL: https://doi.org/10.1190/INT-2016-0160.1 (cit. on p. 22).

Watkins, C. J. C. H. (1989). "Learning from delayed rewards". In: (cit. on p. 8).

Wei, S., O. Yonglin, Z. Qingcai, H. Jiaqiang, et al. (2018). "Unsupervised Machine Learning: K-means Clustering Velocity Semblance Auto-Picking". In: *80th EAGE Conference*. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=92299 (cit. on p. 22).

Wirgin, A. (2004). "The inverse crime". In: *arXiv preprint math-ph/0401050* (cit. on p. 23).

Wu, H. and B. Zhang (2018). "A deep convolutional encoder-decoder neural network in assisting seismic horizon tracking". In: arXiv: 1804.06814 [physics.geo-ph]. URL: http://arxiv.org/abs/1804.06814 (cit. on p. 134).

Xie, S., R. Girshick, P. Dollár, Z. Tu, and K. He (2017). "Aggregated residual transformations for deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1492–1500 (cit. on p. 131).

Xie, X., H. Qin, C. Yu, and L. Liu (2013). "An automatic recognition algorithm for GPR images of RC structure voids". In: *J. Appl. Geophys.* 99, pp. 125–134. ISSN: 0926-9851. DOI: 10.1016/j.jappgeo.2013.02.016. URL: http://www.sciencedirect.com/science/article/pii/S0926985113000487 (cit. on p. 21).

Yilmaz, Ö. (2003). *Seismic data analysis : processing, inversion, and interpretation of seismic data*. en. Society of Exploration Geophysicists. ISBN: 9781560800941. DOI: 10.1190/1.9781560801580. URL: https://market.android.com/details?id=book-kYeioAEACAAJ (cit. on pp. 3, 4).

Youn, H.-S. and C.-C. Chen (2002). "Automatic GPR target detection and clutter reduction using neural network". In: *Ninth International Conference on Ground Penetrating Radar*. Vol. 4758. International Society for Optics and Photonics, pp. 579–583. DOI: 10.1117/12.462229. URL: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/4758/0000/Automatic-GPR-target-detection-and-clutter-reduction-using-neural-network/10.1117/12.462229.short (cit. on p. 133).

Yu, T., D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine (2019). "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning". In: *arXiv preprint arXiv:1910.10897* (cit. on p. 24).

Zhang, L., J. Quieren, and J. Schuelke (2001). "Chapter 10 Self-Organizing Map (SOM) network for tracking horizons and classifying seismic traces". In: *Handbook of Geophysical Exploration: Seismic Exploration*. Ed. by M. M. Poulton. Vol. 30. Pergamon, pp. 155–170. DOI: 10.1016/S0950-1401(01)80024-0. URL: http://www.sciencedirect.com/science/article/pii/S0950140101800240 (cit. on p. 133).

Zhang, Y. and K. V. Paulson (1997). "Magnetotelluric inversion using regularized Hopfield neural networks". In: *Geophys. Prospect.* ISSN: 0016-8025. URL: http://www.earthdoc.org/publication/publicationdetails/?publication=33881 (cit. on p. 20).

Zhao, T., F. Li, and K. Marfurt (2017a). "Constraining self-organizing map facies analysis with stratigraphy: An approach to increase the credibility in automatic seismic facies classification". In: *Interpretation* 5.2, T163–T171. ISSN: 2324-8858. DOI: 10.1190/INT-2016-0132.1. URL: https://doi.org/10.1190/INT-2016-0132.1 (cit. on p. 134).

Zhao, T., J. Zhang, F. Li, and K. Marfurt (2016). "Characterizing a turbidite system in Canterbury Basin, New Zealand, using seismic attributes and distance-preserving self-organizing maps". In: *Interpretation* 4.1, SB79–SB89. ISSN: 2324-8858. DOI: 10.1190/INT-2015-0094.1. URL: https://doi.org/10.1190/INT-2015-0094.1 (cit. on p. 134).

Zhao, X. and J. M. Mendel (1988). "Minimum-variance deconvolution using artificial neural networks". In: *SEG Technical Program Expanded Abstracts*. URL: https://library.seg.org/doi/pdf/10.1190/1.1892433 (cit. on pp. 20, 133).

Zhao, Z. and L. Gross (2017b). "Using supervised machine learning to distinguish microseismic from noise events". In: *SEG Technical Program Expanded Abstracts 2017*. SEG Technical Program Expanded Abstracts. Society of Exploration Geophysicists, pp. 2918–2923. DOI: 10.1190/segam2017-17727697.1. URL: https://doi.org/10.1190/segam2017-17727697.1 (cit. on p. 21).

Zheng, Z. H., P. Kavousi, and H. B. Di (2014). "Multi-attributes and neural network-based fault detection in 3D seismic interpretation". In: *Adv. Mat. Res.* ISSN: 1022-6680. URL: https://www.scientific.net/AMR.838-841.1497 (cit. on pp. 22, 134).

Zhu, J.-Y., T. Park, P. Isola, and A. A. Efros (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232 (cit. on p. 15).

Zhu, W. and G. C. Beroza (2018). "PhaseNet: A Deep-Neural-Network-Based Seismic Arrival Time Picking Method". In: arXiv: 1803.03211 [physics.geo-ph]. URL: http://arxiv.org/abs/1803.03211 (cit. on pp. 21, 134).

Zoph, B., V. Vasudevan, J. Shlens, and Q. V. Le (2018). "Learning transferable architectures for scalable image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710 (cit. on pp. 17, 131).

Zuo, R. and Y. Xiong (2018). "Big Data Analytics of Identifying Geochemical Anomalies Supported by Machine Learning Methods". In: *Nat. Resour. Res.* 27.1, pp. 5–13. ISSN: 1520-7439, 1573-8981. DOI: 10.1007/s11053-017-9357-0. URL: https://doi.org/10.1007/s11053-017-9357-0 (cit. on p. 21).