Microseismic event detection and localization
A migration-based and machine-learning approach using full waveforms

Vinard, N.

DOI
10.4233/uuid:3a065972-1fbc-483a-b9d6-7bc4570d4bef

Publication date
2022

Document Version
Final published version

Citation (APA)
Vinard, N. (2022). *Microseismic event detection and localization: A migration-based and machine-learning approach using full waveforms*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:3a065972-1fbc-483a-b9d6-7bc4570d4bef

Microseismic event detection and localization

A migration-based and machine-learning approach using full waveforms

Microseismic event detection and localization

A migration-based and machine-learning approach using full waveforms

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology,
by the authority of the Rector Magnificus, Prof. Dr. Ir. T.H.J.J. van der Hagen,
chair of the Board for Doctorates,
to be defended publicly on
Wednesday, 21st September 2022 at 15:00 o'clock

by

Nicolas André VINARD

Master of Science in Earth Sciences, Swiss Federal Institute of Technology Zurich
born in Lausanne, Switzerland

Keywords: Machine learning, induced seismicity, microseismic monitoring

Printed by: Gildeprint

Typesetting system: LaTeX.

An electronic version of this dissertation is available at
`http://repository.tudelft.nl/`.

Summary

Microseismic event detection and localization: A migration-based and machine-learning approach using full waveforms

When humans started started exploiting the abundant underground natural resources the Earth has to offer such as hydrocarbons, minerals and heat, we started to experience earthquakes that are related to this exploitation, so called induced earthquakes. Under certain conditions those can damage local infrastructure. However, most events are weak and only sensed by seismic sensors. Microseismic monitoring plays a vital role to optimize and insure the safety of these underground activities and new technologies such as carbon capture and storage.

One key task besides the detection of microseimsic events is to determine the source location of these events using data recorded at the surface. In this thesis we investigate a method to localize weak microseismic events, using a deterministic approach, assuming a dense network of sensors. In simple words this method takes the seismic signals recorded at the Earth's surface and sends them back into the Earth, where the signals start to focus at the point they originated from. This focusing method uses one-way wavefield extrapolation with an estimate of the background velocity model. The advantage of this method is that the weak signals recorded by the different sensors at the surface are amplified as they approach the location of the event that emitted the signal due to constructive interference. However, this is not enough to reliably recover the source location because typically earthquakes do not radiate seismic waves evenly; complex radiation patterns are typically observed depending on the mechanical properties of the rupture. To obtain a strong focused signal at the optimal source location we therefore perform a grid search over possible source mechanisms and increase the strength of the signal by deconvolution. Without taking the source mechanism into account we are not able to obtain accurate source locations, especially at low signal-to-noise ratios. However, by taking the source mechanism into account we are able to retrieve accurate source locations while also retrieving information about the source mechanism. Good results were obtained for 2D synthetic data for both a simple subsurface model as well as the realistic Annerveen salt model even when realistic noise was added.

A drawback of this method is that it is computationally as well as logistically expensive and it requires a dense network of receivers such that backpropagation can be done aliasing free. Therefore, machine learning methods that can deal with much smaller number of sensors and additionally are computationally cheap during the application phase are further explored in the remainder of the thesis.

Deep learning (DL) methods require a large number of training samples. Since for many microseismic monitoring applications there is no field data before the operations start, e.g. fracking, we first train a deep neural network on synthetic data. As a neural network, we consider a convolutional neural network with an architecture similar to the U-Net architecture. This network takes the waveforms as input and outputs a Gaussian distribution in a 3D (x,y,z) volume, representing the source locations at the peak of the 3D distribution. The synthetically trained deep learning algorithm is then applied to hydraulic fracturing field data. For optimal results the synthetic data is augmented with field noise during training. The results show that the neural network reliably localizes the higher magnitude events and thus that synthetic data can be used to train a neural network for the application of field data. Moreover, the locations predicted by the neural network are close to the locations recovered by a field-standard migration-based diffraction stacking method.

While the neural network can reliably estimate the locations of the higher magnitude events it still falls short for most of the low-magnitude events. Therefore, we updated the approach for those events.

To localize the weak microseismic events we fine-tune the neural network parameters with field data. Fine-tuning a neural network is also referred to as transfer learning (TL). Instead of training a DL model from scratch, TL makes use of an existing DL model that was trained for a similar task. This has several advantages such as it not requiring large amounts of training data (since the weights of the model were already optimized with large amounts of data) and a reduced training period. In our case the neural network's weights are updated by training with field data. In addition, we are interested in applying the method for real-time localization. Therefore, we iteratively fine-tune the neural network as new data are recorded. Still, since the event locations migrate over time and appear in different clusters the neural network always lags behind. Therefore, we fine-tune the neural network not only with field data but also with synthetics, which improves the neural network's generalizability. The updated neural network is now capable to accurately localize a large majority of all hydraulic fracturing events given in the data set. Furthermore, the locations recovered by the fine-tuned neural network are more consistent and appear to have more self-consistent depth estimates compared to a diffraction stacking method.

To achieve real-time localization an event first needs to be detected. We explore the possibility of using the output generated by the neural network trained for localization to detect events. Since the network was trained to return a location output given data with an event in the input, it is not expected to return a similar output given noise in the input. Leveraging that idea we build a detector solely using the localization deep learning algorithm. While this detector did return some false detections, it did detect a majority of the events that were also detected by diffraction stacking. Furthermore, it detected events that were not detected by diffraction stacking.

This thesis shows that deep learning algorithms can be pre-trained on synthetic data to localize hydraulic fracturing events. Furthermore, by means of transfer learning the deep learning model is capable of localizing events from waveforms with low signal-to-noise ratios. Finally, the output generated by the deep learning model trained for source-localization can be used as an event detector.

Samenvatting

Opsporing en lokalisatie van microseismische gebeurtenissen: Een op migratie gebaseerde en machinaal-lerende benadering gebruik makende van volledige golfvormen

Toen de mens begon met de exploitatie van de overvloedige ondergrondse natuurlijke rijkdommen die de aarde te bieden heeft, zoals koolwaterstoffen, mineralen en warmte, kregen we te maken met aardbevingen die verband houden met deze exploitatie, de zogeheten geïnduceerde aardbevingen. Onder bepaalde omstandigheden kunnen deze aardbevingen schade toebrengen aan de plaatselijke infrastructuur. De meeste gebeurtenissen zijn echter zwak en worden alleen door seismische sensoren waargenomen. Microseismische monitoring speelt een vitale rol bij het optimaliseren en verzekeren van de veiligheid van deze ondergrondse activiteiten en nieuwe technologieën zoals koolstofafvang en -opslag.

Een belangrijke taak naast het opsporen van microseismische gebeurtenissen is het bepalen van de bronlocatie van deze gebeurtenissen met behulp van gegevens die aan het oppervlak zijn opgenomen. In dit proefschrift onderzoeken we een methode om zwakke microseismische gebeurtenissen te lokaliseren, gebruikmakend van een deterministische benadering, uitgaande van een dicht netwerk van sensoren. Eenvoudig gezegd neemt deze methode de seismische signalen die aan het aardoppervlak zijn opgenomen en zendt ze terug de aarde in, waar de signalen zich gaan concentreren op het punt waar ze vandaan komen. Deze focusmethode maakt gebruik van éénrichtings-golfveldextrapolatie met een schatting van het velocity-depth achtergrondmodel. Het voordeel van deze methode is dat de zwakke signalen die door de verschillende sensoren worden geregistreerd, worden versterkt naarmate zij de plaats naderen van de gebeurtenis die het signaal heeft uitgezonden. Dit is echter niet voldoende om op betrouwbare wijze de plaats van de bron te bepalen, omdat aardbevingen de seismische golf doorgaans niet gelijkmatig uitstralen; er worden doorgaans complexe stralingspatronen waargenomen, afhankelijk van de mechani-

sche eigenschappen van de breuk. Om een sterk gefocusseerd signaal op de juiste bronlocatie te verkrijgen, voeren we daarom een rasterzoektocht uit over mogelijke bronmechanismen en verhogen we de sterkte van het signaal door deconvolutie. Zonder rekening te houden met het bronmechanisme zijn we niet in staat om nauwkeurige bronlocaties te verkrijgen, vooral bij lage signaal-ruis verhoudingen. Door rekening te houden met het bronmechanisme zijn we echter wel in staat om nauwkeurige bronlocaties te verkrijgen, terwijl we ook informatie over het bronmechanisme verkrijgen. Goede resultaten werden verkregen voor 2D synthetische data voor zowel een eenvoudig model van de ondergrond als het realistische Annerveen zout model, zelfs wanneer realistische ruis werd toegevoegd.

Een nadeel van deze methode is dat ze zowel computationeel als logistiek duur is en dat ze een dicht netwerk van ontvangers vereist zodat backpropagatie aliasingvrij kan worden uitgevoerd. Daarom worden in het vervolg van dit proefschrift methoden voor machinaal leren onderzocht die met een veel kleiner aantal sensoren kunnen werken en die bovendien in de toepassingsfase rekenkundig goedkoop zijn.

Deep-learning (DL) methoden vereisen een groot aantal trainingssamples. Aangezien er voor veel microseismische monitoringtoepassingen geen veldgegevens zijn voordat de operaties, bv. fracking, beginnen, trainen we eerst een diep neuraal netwerk op synthetische gegevens. Als neuraal netwerk beschouwen we een convolutioneel neuraal netwerk met een architectuur vergelijkbaar met die van U-Net. Dit netwerk neemt de golfvormen als input en geeft als output een blob in een 3D (x,y,z) volume, die de bronlocaties met maximale waarde weergeeft. Het synthetisch getrainde deep-learning algoritme wordt vervolgens toegepast op de gegevens van het hydraulisch breken van gesteentes in het veld. Voor optimale resultaten worden de synthetische gegevens tijdens de training aangevuld met veldruis. Het resultaat toont aan dat het neurale netwerk op betrouwbare wijze de bronlocatie lokaliseert van de gebeurtenissen met grotere magnitude en dat synthetische gegevens dus kunnen worden gebruikt om een neuraal netwerk te trainen voor de toepassing van veldgegevens. Bovendien liggen de door het neurale netwerk voorspelde locaties dicht bij de locaties die met een huidige standaard migratie-gebaseerde diffractiestapelingsmethode worden teruggevonden.

Hoewel het neurale netwerk de locaties van gebeurtenissen met een grotere magnitude op betrouwbare wijze kan schatten, schiet het nog tekort voor de meeste gebeurtenissen met een geringe magnitude. Daarom hebben wij de aanpak voor die gebeurtenissen bijgewerkt.

Om de zwakke microseismische gebeurtenissen te lokaliseren stemmen we de neurale netwerkparameters af met veldgegevens. Het afstemmen van een neuraal netwerk wordt ook wel transfer learning (TL) genoemd. In plaats van een DL-model vanaf nul te trainen, maakt TL gebruik van een bestaand DL-model dat voor een soortgelijke taak is getraind. Dit heeft verschillende voordelen, zoals het feit dat er geen grote hoeveelheden trainingsgegevens nodig zijn (omdat de gewichten van het model al met grote hoeveelheden gegevens zijn geoptimaliseerd) en een kortere trainingsperiode. In ons geval worden de gewichten van het neurale netwerk geactualiseerd door te trainen met veldgegevens. Bovendien zijn wij geïnteresseerd in toepassing van de methode voor real-time lokalisatie. Daarom stellen wij het neurale

netwerk iteratief bij naarmate nieuwe gegevens worden geregistreerd. Aangezien de locaties van de gebeurtenissen in de loop van de tijd migreren en in verschillende clusters voorkomen, loopt het neurale netwerk echter altijd achter de feiten aan. Daarom stemmen we het neurale netwerk niet alleen af met veldgegevens, maar ook met synthetische gegevens, wat de generaliseerbaarheid van het neurale netwerk verbetert. Het bijgewerkte neurale netwerk is nu in staat om een grote meerderheid van alle gebeurtenissen als gevolg van hydraulisch breken in de dataset nauwkeurig te lokaliseren. Bovendien zijn de door het fijnafgestemde neurale netwerk gevonden locaties consistenter en hebben ze betere diepteschattingen in vergelijking met een diffractie-stapelingmethode.

Om real-time lokalisatie te bereiken moet een gebeurtenis eerst worden gedetecteerd. Wij onderzoeken de mogelijkheid om direct gebruik te maken van de output die wordt gegenereerd door het neurale netwerk dat voor lokalisatie is getraind. Aangezien het netwerk getraind is om een lokalisatieoutput te genereren met data met een gebeurtenis in de input, wordt niet verwacht dat het een gelijkaardige output zal genereren met ruis in de input. Gebruikmakend van dat idee bouwen we een detector die uitsluitend gebruik maakt van het lokalisatie deep-learning algoritme. Hoewel deze detector enkele valse detecties teruggaf, detecteerde hij een meerderheid van de gebeurtenissen die ook werden gedetecteerd door de diffractie-stapelingsmethode. Bovendien detecteerde hij gebeurtenissen die niet werden gedetecteerd door diffractie-stapelingsmethode.

Deze dissertatie toont aan dat deep-learning algoritmen vooraf getraind kunnen worden op synthetische data om gebeurtenissen bij het hydraulisch breken van gesteentes te lokaliseren. Verder is het deep-learning model door middel van transfer learning in staat om gebeurtenissen te lokaliseren uit golfvormen met lage signaal-ruis verhoudingen. Tenslotte kan de output van het deep-learning model, getraind voor bronlokalisatie, gebruikt worden als een event detector.

Contents

1 Introduction

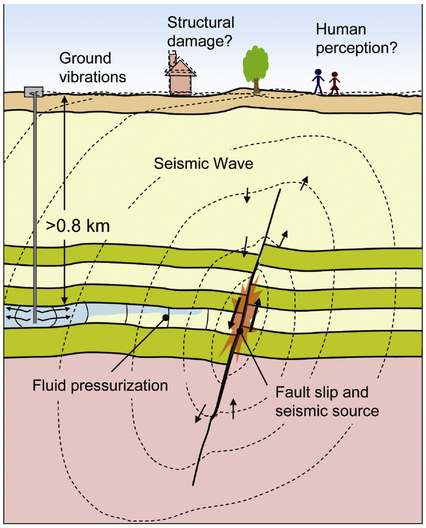
"Earthquakes travelling through the interior of the globe are like so many messengers sent out to explore a new land. The messages are constantly coming and seismologists are fast learning to read them. "

Reginald Aldworth Daly, 1926

1.1 Induced Seismicity

Induced seismicity, i.e. seismicity arising due to industrial activities, has been researched for over a hundred years with its origins in the mining industry [*McGarr et al.*, 2002]. However, the interest in the topic has gained a lot of attention in the past few decades due to major induced earthquakes around the globe that directly impacted thousands of people's lives. Some industrial activities associated with induced seismicity include water reservoirs, mining, wastewater injection, enhanced geothermal systems, carbon sequestration, hydrocarbon extraction and hydraulic fracturing. During such activities many small earthquakes that can not be felt at the surface are occurring. Such earthquakes are called microseismic events. Those events are also closely monitored as they can provide information about the locations of fractures in the subsurface and in order for the operators to prove that the magnitude of the events are within the limits imposed by regulators.

The illustration in Fig. 1.1 shows a schematic of an earthquake triggered by fluid injection and an example of what seismic data recorded along an array of geophones from an induced event looks like. Note that the fault-slip source mechanism creates directivity effects on the recorded data, visible, e.g. as amplitude variations in the first arrival. In the next few paragraphs the industrial activities mentioned above are briefly introduced.



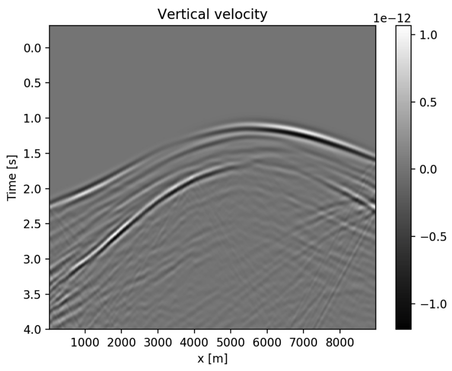


Figure 1.1: *Left: Illustration of induced earthquake due to fluid injection by Rutqvist et al. [2014] and right: synthetic example of seismic earthquake data.*

Water-reservoir triggered earthquakes may not be the first example that comes to mind when thinking about the drivers behind industry-related earthquakes. However, there have been several reports of reservoir-triggered earthquakes with moment magnitudes (**M**) greater than 6, such as in Greece in 1966 and in Koyna, India in 1967 [*Gupta*, 2002]. The earthquake in Koyna resulted in the deaths of nearly 200 people and thousands of injured and homeless. The main driver behind this type of seismicity is the extra mass of water increasing the stress regime in that region. However, different mechanisms can be involved, which lead to these high seismic activities [*Gupta*, 2002].

Monitoring mining activities can help identify unstable locations and the location of rockbursts, which is crucial for a rescue mission [*Mendecki*, 1993; *Mendecki et al.*, 1999].

Wastewater injection induced seismicity is particularly prominent in the central and eastern United States, with earthquakes reaching moment magnitudes up to 5.7 [*Keranen et al.*, 2014]. The earthquake reaching **M** 5.7 in Oklahoma destroyed homes and damaged infrastructure. A study determined that the initial rupture plane of the series of earthquakes that led to the **M** 5.7 earthquake was within 200 m from an active injection well [*Keranen et al.*, 2013].

Enhanced geothermal systems struggle with being widely accepted for use - albeit being one of the technologies that can become part of the transition to a hydrocarbon-free society - due to their history of inducing earthquakes. Before being able to use the heat stored at depths between 2 to 5 km (depending on the geothermal gradient in the region) these systems require a permeable subsurface for the fluids to circulate. In many situations this is not naturally present and thus the permeability needs to be artificially increased. This is achieved by injecting fluids at high pressure and thereby create fractures [*Majer et al.*, 2007]. Once the permeability is increased this allows the injected fluids to circulate and heat up before pumping them back up to the surface. However, both during the creation of the fractures and in production, microseismic events can be induced. Therefore, it is important to understand how larger events can be avoided to ensure the realization of enhanced geothermal projects, avoiding situations such as in Basel, Switzerland, where the entire project had to be ceased due to the occurrence of a **M** 3.4 earthquake [*Deichmann and Giardini*, 2009].

Carbon sequestration, also known as carbon capture an storage, is a technology that could play a key role towards reducing greenhouse gas emissions. The aim of carbon sequestration is to permanently store CO_2 in the subsurface. As with all fluid injection projects, carbon sequestration also faces the challenge of induced seismicity. The CO_2 capture projects to date did not lead to seismicity that can be felt at the Earth's surface. However, as projects are scaled up for commercial use, the history from other injection-related projects suggests that the issue of induced seismicity needs to be carefully evaluated for carbon sequestration projects as well [*Nicol et al.*, 2011; *Zoback and Gorelick*, 2012; *Rutqvist et al.*, 2016].

In the extraction of hydrocarbons there actually are very few reports of induced seismicity compared to the total number of existing fields. However, this can also be due to a lack of monitoring systems [*Suckale*, 2010]. For fields that are far offshore or in deserted areas the resulting seismicity has most likely no impact to the general public. However, as similar operations are performed closer to urban areas the consequences of the seismic activity are much more severe. A perfect example is found in one of the most densely populated countries in Europe that also happens to possess Europe's largest inland gas field, the Netherlands. The main driver behind the induced seismicity in the Groningen gas field is reservoir compaction [*Van Wees et al.*, 2014], along with soft soils near the surface amplifying the effect. This led to an earthquake with local magnitude (M_L) 3.6 in 2012 that damaged local infrastructure.

Finally, hydraulic fracturing also aims at creating fractures, as in enhanced geothermal systems, however for the purpose of extracting hydrocarbons from unconventional reservoirs such as shales. Most of these events are directly related to the fracturing activity, which lead to weak microseismic events. However, there have also been numerous cases of felt seismicity in the United Kingdom, Canada, the United States and China induced by hydraulic fracturing operations due to faults near the injection sites [*Schultz et al.*, 2020a].

The induced seismicity led governments to mandate strict regulations for these activities to mitigate induced seismicity, which also include the use of traffic light protocols (TLPs). TLPs contain thresholds on the magnitudes of the detected events to decide when the activities can proceed as normal (green), mitigation measures must be started (amber) and when operations must be ceased (red) [*Schultz et al.*, 2020b]. To monitor risks associated to the activities, microseismic monitoring systems must be set up, which are capable to detect, localize and retrieve the magnitudes of the weak microseismic events. The seismic activity is recorded by seismic sensors that can be deployed in deep boreholes or close to the surface. The sensors are often placed close to the surface, as this is cheaper, and in case of a defect the sensors can more easily be exchanged or repaired. For optimal monitoring, events should be detected, their magnitudes and locations determined and their source mechanism computed. In this thesis we focus primarily on the source localization and secondly on the detection.

1.2 Localization methods

As stated above, identifying the location of microseismic events is an important topic. A main challenge remains that the recorded events are weak with signal-to-noise ratios below 1. Fig. 1.2 shows a normal moveout-corrected gather of a weak microseismic event with the first arrivals highlighted by red arrows.

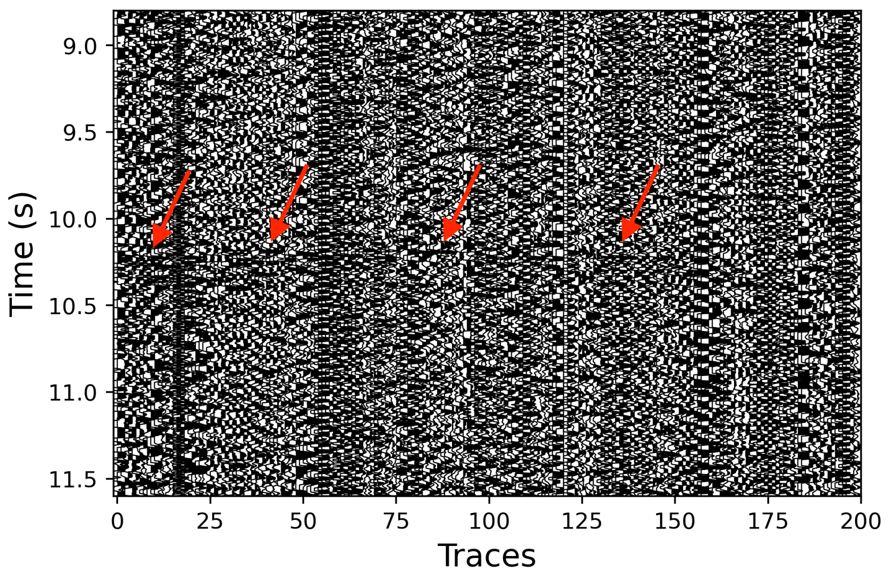


Figure 1.2: *Normal moveout-corrected weak microseismic event recorded during hydraulic fracturing in the Barnett Shale, Fort Worth basin, Texas. Red arrows highlighting first arrivals.*

The signals recorded by such low-magnitude events can hardly be detected using only a few sensors. However, when many sensors are available, different array and waveform based processing methods can be used to detect the signal by adding up the signals recorded at the individual sensors.

One commonly used migration-based detection and location method is based on diffraction stacking [*Duncan and Eisner*, 2010; *Chambers et al.*, 2010; *Gharti et al.*, 2010]. In diffraction stacking the amplitudes of the individual waveforms are stacked along the expected moveout curve (i.e. relative traveltime arrivals) assuming the subsurface velocity model is roughly known. Since the moveout curve depends on the location of the event, which is unknown, the amplitudes are stacked along different precomputed moveout curves from different hypothetical source-locations. By finding the stack with the highest amplitude, the correct source location can be identified.

However, microseismic events, similar to earthquakes, do not have a symmetric radiation pattern. Thus, the complex radiation pattern of microseismic events requires some corrections prior to stacking to avoid summing of amplitudes with opposite polarities. The better approaches take the radiation pattern into account. In these approaches, in addition to considering the moveout from every possible source location, the moment tensor is considered for every source point. This information is used to correct the polarity of the amplitudes prior to stacking. The highest stack is then found at the correct source location and correct moment tensor [*Rodriguez et al.*, 2012; *Anikiev et al.*, 2014; *Chambers et al.*, 2014; *Staněk et al.*, 2015; *Zhebel and Eisner*, 2015].

In other similar approaches, the recorded seismic signals are numerically extrapolated backwards in time, thereby generating an image of the seismic source in time and space. Imagine for example a water tank surrounded by transducers on its boundary that record the acoustic waves that propagate in water. If you throw a stone-pebble into the middle of the quiet water tank it will create ripples that travel from the middle towards the boundaries, where the transducers record the pressure change over time. Once the water tank is quiet again, we can time-reverse the recorded signals at each transducer and have each transducer emit that signal. A chaotic wave pattern will appear but as time moves on you will observe how the chaotic pattern starts to form the ripples that you observed when you threw the stone into the tank, with the only beautiful exception that the ripples seem to be moving back in time towards the location where the stone first touched the surface of the water. This is the time-reversal technique [*Fink*, 1992], as, e.g., used in reverse-time migration [*Whitmore*, 1983; *Loewenthal and Mufti*, 1983; *Baysal et al.*, 1983]. Luckily, the same can be achieved using numerical simulations and there is no need to back inject an earthquake from observations at the Earth's surface.

In seismology this concept was first introduced as a synthetic study in 1982 in 2D using a finite-difference approach to both forward model the source to generate synthetic data and to then time-reverse the signals recorded at each receiver and back propagate it using the time-reversed seismograms as input [*McMechan*, 1982]. Soon after, this principle was applied on field data using the 2D approximation [*McMechan et al.*, 1985; *Rietbrock and Scherbaum*, 1994], extended to 3D [*McMechan et al.*, 1988; *Chang and McMechan*, 1991] and to the elastic 2D case [*Hu and McMechan*, 1988]. Various numerical and field studies thereof at exploration scale have been investigated in the more recent past [*Gajewski and Tessmer*, 2005; *Xuan and Sava*, 2010; *Artman et al.*, 2010; *Bazargani and Snieder*, 2013; *Li et al.*, 2019b; *Yang and Zhu*, 2019].

With the rapid advancement of machine learning methods across a wide spectrum of fields, it is no wonder that it found its way to seismology. Various deep learning based earthquake detection methods [*Wu et al.*, 2018b; *Meier et al.*, 2019; *Dokht et al.*, 2019; *Mousavi et al.*, 2019] and event clustering and detection methods [*Perol et al.*, 2018] have been investigated. Similarly deep learning methods have been applied and developed for the sake of source localization from the seismic signals, some of which are based on arrival time picking [*Ross et al.*, 2018a,b; *Meier et al.*, 2019; *Zhou et al.*, 2019; *Zhu and Beroza*, 2019; *Chai et al.*, 2020; *Zhu et al.*, 2019; *Zhang et al.*, 2020a] and others that directly retrieve the location given the seismic waveforms as input [*Kriegerowski et al.*, 2019; *Zhang et al.*, 2020b; *Van den Ende and Ampuero*, 2020; *Mousavi and Beroza*, 2020a]. However, main challenges are (1) the lack of available data for training, (2) that the data are not labeled (which is a requirement for supervised machine learning methods) and (3) unbalanced data sets, e.g. lack of high magnitude events in the training set could lead to large magnitude events not being detected by a neural network trained for earthquake detection as reported by [*Mousavi and Beroza*, 2020b].

1.3 Research questions and outline

In this thesis we investigate both migration based as well as machine learning based source detection and localization methods. In particular we seek to answer

- How one-way inverse wavefield extrapolation operators can be used for the localization and source mechanism estimation of induced earthquakes;
- Whether deep neural networks can be trained on synthetic data to be applied on field data in particular for situations of scarce field data availability;
- How weak microseismic events can accurately be localized by fine-tuning a neural network trained on synthetic data;
- How events can be detected in real-time in order to combine the detection with the localization neural network for real-time microseismic monitoring.

The following chapters are part of this thesis:

- Chapter 2: *Data-driven earthquake localization and source-mechanism estimation based on wavefield extrapolation and 2D deconvolution in high-noise environments.*

  In this chapter a method to localize weak microseismic events is investigated which requires a dense network of sensors at the surface. The dense network together with some knowledge about the subsurface, i.e. the seismic velocities that determine how the seismic waves propagate through the Earth, and about the physics of wave propagation, we can use a method which allows us to numerically backpropagate seismic waves back into the subsurface. This concept is employed in many different source localization applications. The method is computationally as well as logistically expensive as it requires many sensors. As cost plays an important role for companies, most often much fewer sensors are used in practice. Thus, in chapters 3 and 4 we consider a method that can recover the source locations with fewer sensors.

- Chapter 3: *Localizing microseismic events on field data using a U-Net based convolutional neural network trained on synthetic data.*

  In this chapter machine learning, in particular deep learning, is used to determine the locations of microseismic events recorded during hydraulic fracturing operations, given seismic data as input. The source location is returned as a 3D Gaussian distribution where its peak defines the location. A main limitation of many deep learning methods is that they require large data sets to train. The hydraulic fracturing field data set used to demonstrate the applicability of the method is too small to train a deep learning model from scratch. Thus, the main purpose of this chapter is to demonstrate how a deep learning model can (1) be trained on synthetic data to (2) be applied on field data to recover the locations of microseismic events.

- Chapter 4: *Localizing weak microseismic events using transfer learning with a deep neural network.*

  In this chapter the issue of continuous microseismic source localization is addressed. As a basis we start with the deep learning model that was trained on synthetic data. This step could be performed before a microseismic monitoring system is set up. Next, we want to use the deep learning model for seismic source localization as soon as possible after the microseismic monitoring system is in place and recording. We achieve this by optimizing the deep learning model with field data collected at the end of each day. This way the model constantly improves and is able to return better source locations comparable to locations recovered by traditional state-of-the-art methods, such as diffraction stacking methods. One advantage of this method is that it achieves those results using data recorded by fewer sensors. One remaining disadvantage is that while the method returns the source locations it still relies on other methods that first detect whether a signal is present in the data or not.

- Chapter 5: *Fine tuning a deep neural network to localize low magnitude earthquakes.*

  In this chapter the possibility of fine-tuning the synthetically trained deep learning using high magnitude field data in order to localize weaker magnitude events is investigated. These initial results are promising, where iteratively fine-tuning the model's weights using increasingly lower magnitude events can result in a network model with enhanced feature extraction capabilities for localizing weaker magnitude events not included in the training set.

- Chapter 6: *Detecting microseismic events using a deep neural network trained for localization.*

In this chapter the possibility of using the deep learning model trained for source localization to act as a microseismic event detector is explored. Since the deep learning model is trained to return a 3D Gaussian distribution given microseismic data as input it returns some information about how reliable it is at recognizing seismic waveforms in the data in terms of the maximum amplitude in its output. If the input only contains noise, ideally the deep learning model would return a 3D output full of zeros, because there is no relevant information for it to extract. This is explored on three hours of continuous field data. An efficient detector should (1) not miss any events and (2) not cause false alarms. We developed a strategy that takes those two points into account and uses the deep learning model trained for localization. In this way, we established a framework for real-time microseismic event detection and localization using a single deep learning model.

- Chapter 7: *Conclusions and recommendations.*

  Finally, in this last chapter conclusions and recommendations of the previous research chapters are discussed.

2

Data-driven earthquake localization and source-mechanism estimation based on wavefield extrapolation and 2D deconvolution in high-noise environments

"Nobody ever figures out what life is all about, and it doesn't matter. Explore the world. Nearly everything is really interesting if you go into it deeply enough."

Richard P. Feynman

Abstract Near real-time detection and localization of weak induced earthquakes is a topic of active research in the field of microseismic monitoring with the purpose of reducing the seismic risk and increasing the productivity of a reservoir. At low signal-to-noise ratios standard earthquake detection methods fail to detect microseismic signals. Thus, over the past few decades seismologists started using data-driven methods to detect and image earthquakes recorded with spatially Nyquist-sampled arrays. This is achieved by exploiting the time-invariance of the wave equation in a lossless medium, which allows the extrapolation of the wavefields back towards the source they originated from. In practice it is possible to apply these methods to media such as the Earth, which are not loss-free. Applying

Parts of this chapter are part of the conference proceeding: N. A. Vinard, G. G. Drijkoningen and D. J. Verschuur (2019), Data-driven earthquake detection, localization and source mechanism estimation based on wavefield extrapolation and 2D deconvolution in high noise environments, Annual Meeting Seismological Society of America Technical Sessions, pp. 963-694.
Note that minor changes have been introduced to make the text consistent with the other chapters.

those methods lead to increased signal-to-noise ratios, which naturally result in an increased detection threshold.

In this chapter we downward continue the wavefields in depth with the weighted-least squares (WLSQ) one-way acoustic wavefield extrapolation operator in the space-frequency domain. These operators were designed to work efficiently and with high accuracy in heterogeneous media. Ideally, the amplitudes of the extrapolated wavefields reach a maximum at the location of the source. This can be used as an imaging condition to determine the hypocenter location and origin time. However, due to the radiation pattern of the source, there are polarity changes in the P- and S-wave phases along the receiver array, which lead to destructive interference of the extrapolated data at the source. We propose to maximize the defocused signal by deconvolving the extrapolated data by an ideal filter. The ideal filter represents the solution of the surface wavefield downward extrapolated at the source depth for a source mechanism matching the recorded wavefield. Since the source mechanism and location is unknown, a database of focused wavefields for different source mechanisms and locations, given the velocity model and the acquisition geometry as input, is precomputed. The maximum amplitude is reached when the source mechanism of the data matches the filters source mechanism and location the most. With this procedure the hypocenter, origin time and an estimate of the source mechanism can be recovered simultaneously.

2.1 Introduction

Large-scale industrial activities in the subsurface can induce earthquakes. These are especially problematic if they occur close to urban areas and result in an increased seismic hazard in that region. Induced earthquakes can occur from a large variety of subsurface activities such as waste-water injection [*Ellsworth*, 2013; *Keranen et al.*, 2014], enhanced geothermal systems [*Majer et al.*, 2007] or the depletion of large oil and gas fields [*Vlek*, 2018]. The North of the Netherlands is experiencing induced earthquakes due to the depletion of the Groningen gasfield, Europe's largest inland gas field. After being discovered in 1959 from the drilling of the Slochteren-1 well, the production started in 1963 and until 2015 2115 billion m^3 of the natural gas had been extracted [*van Thienen-Visser and Breunese*, 2015]. As a consequence, seismic events in the North of the Netherlands started to be detected since 1986. Thus, the seismic network in that region was expanded in the following years, especially since this part of the Netherlands was considered a seismically inactive region.

In usually seismically quiet regions such as the Netherlands, the local infrastructure is often not designed to withstand strong ground motion. Furthermore, soft soils and the shallow depth of the induced earthquakes increase the risk of damages in that area. The largest magnitude event was recorded on the 16th of August 2012 with a local magnitude (M_L) of 3.6. Due to the increased seismic activity and damages caused in the past years the Dutch government decided to regulate and eventually halt production by 2022.

Passive microseismic monitoring plays an important role to better understand the situation and how it might evolve in the future. Microseismic monitoring is typically employed to inspect the seismic activity in the region of interest. The main tasks of microseismic monitoring are event detection, event magnitude estimation, source-location identification and source-mechanism inversion. The ability to successfully locate and determine the source mechanism of small earthquakes adds valuable information to locally study (1) the possibility that larger earthquakes have small precursors (2) the subsurface with additional data and (3) the active fault region. For the detection, localization and source-mechanism inversion of events at low signal-to-noise ratios (S/N) migration-type methods are typically employed. These can be divided into traveltime stacking and reverse-time modeling methods [*Zhebel and Eisner*, 2015]. In this work we focus on the latter.

The application of migration-type methods for determination of source parameters dates back to 1982 [*McMechan*, 1982]. Inspired by reverse-time migration (RTM) [*Whitmore*, 1983; *Loewenthal and Mufti*, 1983; *Baysal et al.*, 1983], McMechan time reverses the seismograms and inputs the time-reversed wavefield as source into a two-way acoustic wave-equation finite-difference modeling solver. The difference to active seismics being that the origin time, location and source mechanism are unknown. Therefore, a different imaging condition is needed. In his synthetic study *McMechan* [1982] determines the origin time and location of the source by seeking the maximum acoustic amplitude. By numerically back propagating the recorded wavefields at the surface through a subsurface velocity model the surface noise is attenuated whilst the signal is constructively interfering as the wavefield moves closer to the region it originated from given that an accurate velocity model is used. A few years later this method was successfully applied to field data [*McMechan et al.*, 1985; *Rietbrock and Scherbaum*, 1994]. *Hu and McMechan* [1988] extended the method to the elastic wave equation presenting possibilities to get the location, extent, orientation and radiation pattern of the event. As the origin time of the source is unknown *McMechan* [1982] used the maximum amplitude as an imaging condition. If both P- and S-waves are used, a better imaging condition can be used based on the fact that the extrapolated P- and S-waves will focus at the origin time and thus the correlation of P- and S-wave energy can be used as imaging condition [*Artman et al.*, 2010], given that accurate S- and P-wave models are used.

Due to the complexity of the source mechanism the surface-recorded wavefields show differences in polarities. This can lead to destructive interference in the extrapolated wavefields, and thus, lower the amplitude of the focused wavefield. To overcome this issue the unknown source mechanism can be taken into account. The source mechanism contains valuable information about the mechanism behind the earthquake and thus, it is desirable to retrieve these parameters as well. This is for instance done in traveltime stacking where prior to stacking the polarities are corrected considering the effect of the source mechanism [*Anikiev et al.*, 2014; *Chambers et al.*, 2014]. *Xuan and Sava* [2010] use time-reversal imaging solving the acoustic and elastic two-way wave equation and address issues bound to the method such as inadequate acquisition geometries, errors in the velocity model and noise. They approach the problem of the unknown source mechanism by constructing ideal solutions of the focused wavefields at various locations given the acquisition geometry and velocity model. Using this database of focused wavefields they compute the crosscorrelation between the wavefields in their database and the extrapolated synthetic wavefield at different time steps. The maximum of the crosscorrelation indicates a likely location and origin time of the earthquake.

Our objective is to accurately estimate the source parameters, i.e. the hypocenter (location and origin time) and source mechanism, of induced earthquakes under the presence of strong correlated noise and a complex velocity model. We achieve this by using the one-way acoustic wave equation to extrapolate the wavefields recorded at the surface to different depths. One drawback of using one-way wave equations is that turning waves cannot be handled because around the turning point up- and downgoing waves are coupled [*Wapenaar and Berkhout*, 1985]. Our working hypothesis is that this should not be an issue because we are interested in imaging local earthquakes occurring underneath the array. We use one-way wavefield extrapolation operators that can handle heterogeneous media, based on weighted least squares (WLSQ) optimization [*Thorbecke et al.*, 2004]. In laterally invariant media, extrapolation operators based on the phase-shift operator [*Gazdag*, 1978] can be applied. We also take the source mechanism into account by designing a set of optimal filters and we compute the deconvolution by deconvolving the inverse extrapolated surface wavevields by the optimal filters, which boosts the extrapolated signal when the location and type of source match well with the corresponding filter. This is a similar approach to *Xuan and Sava* [2010]. Indeed, if the deconvolution filter matches the type of source and extrapolation depth at the source, the correct source location and estimate of the source mechanism is found. The differences to *Xuan and Sava* [2010] are in the usage of different operators for backpropagation (two-way vs one-way extrapolation), cross-correlation vs deconvolution, differences in the complexity of the velocity models (layered vs laterally varying media), the noise added to the synthetics (synthetic Gaussian noise vs correlated field noise), scarce vs dense receiver sampling and dense vs coarse filter sampling. In this chapter we only consider a 2D scenario, i.e. with 2D forward modelling and 2D inverse wavefield extrapolation.

In the following we will describe the methodology, which includes the WLSQ one-way wavefield extrapolation operator, the creation of the optimal filters and the workflow of the localization and source-mechanism estimation method. Next, results are presented starting with a simple layered 2D model and finally with a more complex 2D model and strong superimposed field noise. Finally, we end this chapter with a discussion and conclusions.

2.2 Theory and Methodology

We briefly describe inverse wavefield extrapolation and the WLSQ inverse wavefield extrapolator [*Thorbecke et al.*, 2004]. Next, we describe the source-localization workflow, which includes the design of the filters, inverse wavefield extrapolation and deconvolution.

2.2.1 Inverse wavefield extrapolation

Seismometers on the Earth's surface record the seismic waves emitted by earthquakes. The recorded wavefields can be extrapolated in depth back towards the source they originated from. This process is called inverse wavefield extrapolation. To derive forward extrapolation operators, the causal solution of the Green's function is used [*Wapenaar and Berkhout*, 1989]. Inverse wavefield extrapolation operators on the other hand are based on the anti-causal part implied by the two-way wave equation.

In a homogeneous medium the forward extrapolation operator can be computed analytically, as described in more detail in Appendix A and this appendix also includes the definitions of the forward and backward Fourier integrals. Here, the theory is presented for a 3D setting, however, only the 2D case is considered in the examples of this chapter. In the frequency-wavenumber domain (ω, k_x, k_y), the downward-forward extrapolation operator, $\tilde{W}^+$, is defined as,

$$\begin{aligned}\tilde{W}^+(k_x, k_y, \omega, \Delta z) &= e^{-jk_z\Delta z} \\ &= \begin{cases} e^{-j\Delta z\sqrt{k^2-(k_x^2+k_y^2)}} & \text{for } k_x^2+k_y^2 \leq k^2 \text{ propagating area} \\ e^{-\Delta z\sqrt{(k_x^2+k_y^2)-k^2}} & \text{for } k_x^2+k_y^2 > k^2 \text{ evanescent area,} \end{cases}\end{aligned} \tag{2.2.1}$$

with j being the imaginary unit, Δz the chosen extrapolation step in depth and $k = \omega/c$, with the wave propagation velocity c. Eq. 2.2.1 is known as the phase-shift operator *Gazdag* [1978]. Note that temporal Fourier transforms are denoted by capital letters and spatial transforms by a tilde. From this equation it can be inferred that the amplitude spectrum in the propagating wavenumber area is constant and equal to 1 and that it is exponentially decaying in the evanescent wavenumber area. A stable inverse wavefield extrapolation operator, $\tilde{F}$, can be computed by taking the complex conjugate of eq. 2.2.1 [*Wapenaar and Berkhout*, 1989]:

$$\tilde{F}(\Delta z) = \left[\tilde{W}(\Delta z)\right]^* = \begin{cases} e^{j\Delta z\sqrt{k^2-(k_x^2+k_y^2)}} & \text{for } k_x^2+k_y^2 \leq k^2 \\ e^{-\Delta z\sqrt{(k_x^2+k_y^2)-k^2)}} & \text{for } k_x^2+k_y^2 > k^2. \end{cases} \tag{2.2.2}$$

The complex conjugate only acts on the imaginary part of the operator, thus leaving the real part, i.e. the evanescent spectrum, decaying. If one were to take the direct inverse of eq. 2.2.1, the evanescent term would blow up. For further details and the derivation of inverse wavefield extrapolation operators in vertically and laterally inhomogeneous media, we refer the reader to *Wapenaar and Berkhout* [1989].

■ 2.2.2 WLSQ one-way wavefield extrapolation operators

To handle heterogeneity, the extrapolation is carried out in the space-frequency domain, rather than the wavenumber-frequency domain. Thus, the phase-shift operator is Fourier transformed to the space-frequency domain. The resulting equation is then discretized and truncated to a finite number of points. The accuracy of the operators can be assessed by comparing this truncated operator with the analytical phase-shift operator in the wavenumber spectrum [*Thorbecke et al.*, 2004]. To find the optimal operator, a WLSQ optimization is implemented. The seeked optimized operator should be equal to the ideal phase-shift operator for the propagating waves. However, in the evanescent domain it is not restricted to equal the exact phase-shift operator, which is referred to as smooth phase-shift operator [*Thorbecke et al.*, 2004]. In the smooth phase-shift operator the amplitude and phase for wavenumbers outside the band of interest are designed to smoothly decay to zero, hence its name. The maximum angle of propagation is also included in the definition of the smooth phase-shift operator. This results in an extrapolation operator that - given the limited size in the spatial coordinates - is much more accurate than other truncated operators, as discussed in *Thorbecke et al.* [2004].

The weighted least squares (WLSQ) one-way extrapolation operators are applied as spatial convolution operators in the space-frequency domain. Spatial convolution is an expensive computation, which grows quadratically with the size of the kernel [*Wu et al.*, 2018a]. Extrapolation of wavefields in heterogeneous media requires spatially short operators in order to take into account lateral velocity variations. However, to handle large propagation angles a spatially larger operator is required. Additionally, since the extrapolation is carried out recursively over many depth steps it is important that the amplitudes for all wavenumbers of the operator stay below a given threshold to keep the operation stable [*Thorbecke et al.*, 2004]. The WLSQ one-way wavefield extrapolation operators take these points into account and they were designed to work efficiently and with high accuracy in heterogeneous media.

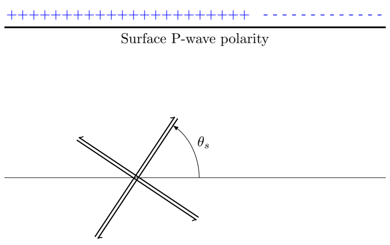


Figure 2.1: *Example of double couple source with inclination θ_s from the horizontal and its corresponding P-wave polarity at the surface (plus and minus signs).*

2.2.3 Workflow to retrieve source parameters

To recover the source location of a seismic event we inverse extrapolate the surface wavefields in depth using the afore-mentioned WLSQ one-way extrapolation operators [*Thorbecke et al.*, 2004]. The surface wavefields need to be extrapolated to several depths since the source depth is unknown. If the wavefields are extrapolated to the correct source depth, the energy should focus, indicating the location and origin time of the source. However, opposing polarities along the recorded events due to complex source mechanisms might weaken the focusing at the depth of the source. Furthermore, if strong noise is present in the data, the focused wavefield may still have lower amplitude than the noise.

In this 2D synthetic study we model the induced events as double-couple (DC) sources, which represent a shear dislocation, i.e. a rupture along a fault. In 2D the DC mechanism can be described by the inclination angle from the horizontal, θ_s. The inclination angle of a double-couple source from the horizontal as well as its corresponding P-wave polarity at the surface is illustrated in Fig. 2.1.

To further boost the focused signal we deconvolve the inverse extrapolated wavefields by a set of ideal filters. To create the filters we model several double-couple sources from different locations within the region where events are expected using a known velocity model. The elastic surface wavefields are modeled with the software Salvus [*Afanasiev et al.*, 2019a]. To create a filter, the detected surface wavefield, s, from a source located at $(x_s,\ z_s)$ is inverse extrapolated using the extrapolation operator, F, to its corresponding source depth, z_s, and finally a window, $\mathcal{W}$, around the focused wavefield is taken and saved to the filter database. A single filter is thus obtained as,

$$h(x,t;z_s,\theta_s) = \mathcal{W}(x,t;x_s)\left\{\mathcal{F}_t^{-1}[S(x,z=z_0,\omega;x_s,\theta_s) * F(x,z_0-z_s,\omega)]\right\}, \quad (2.2.3)$$

where $\mathcal{F}_t^{-1}$ describes the temporal inverse Fourier transform and $*$ stands for the spatial convolution operator. For each modelled source eq. 2.2.3 is applied twice, since the z-component of the wavefield typically has a stronger P-wave signal compared to the x-component, which has a stronger S-wave signal. Thus, the vertical surface wavefield is inverse extrapolated with the P-velocity model to obtain the filter h_z^P and the horizontal wavefield is inverse extrapolated with the S-velocity model to obtain filter h_x^S. Note that the subscripts x and z denote the horizontal and vertical wavefield, respectively and the superscripts P and S denote inverse extrapolation with the P- and S-velocity model, respectively.

Here, the inverse wavefield extrapolation operator works under the assumption that the P- and S-waves propagate to the surface as the same wave mode. Thus, in this chapter converted waves are not dealt with during inverse wavefield extrapolation, as we make use of single-mode operators. In principle, converted waves could be dealt with, although then inverse wavefield operators with similar mode-conversions would be needed.

Similarly, to recover the source location and mechanism of an earthquake, the vertical wavefields are inverse extrapolated with the P-wave velocity model and horizontal wavefields are inverse extrapolated with the S-wave velocity model. Furthermore, since the source location is unknown the vertical and horizontal surface wavefields (d_z and d_x, respectively) are inverse extrapolated to various depths. The downward continued surface wavefields are described as d_z^P and d_x^S.

Once we have the filter database and the inverse extrapolated P- and S-wavefields at various depths, we can follow the workflow summarized in Fig. 2.2, which will be explained next. The deconvolution is computed in the frequency-wavenumber domain and the result is then transformed back to the space-time domain by inverse Fourier transform as,

$$d_{dec}(x,z,t) = \tilde{\mathcal{F}}_{tx}^{-1}\left(\frac{\tilde{D}(k_x,z,\omega)\tilde{H}^*(k_x,\omega;z_s,\theta_s)}{\tilde{H}(k_x,\omega;z_s,\theta_s)\tilde{H}^*(k_x,\omega;z_s,\theta_s)+\epsilon}\right), \tag{2.2.4}$$

with $\tilde{D}$ the inverse extrapolated wavefield, d_{dec} the data after deconvolution by one of the filters, H, in the database and $\tilde{\mathcal{F}}_{tx}^{-1}$ standing for the temporal and spatial inverse Fourier transform. The complex conjugate is denoted by the star and ϵ is the stability factor used to avoid division by zero.

As we do not know the filter location and type of source mechanism in advance, a loop over all filters in the database is performed at each inverse extrapolation step. The individual filters are defined as $H_{x,i}^S$ and $H_{z,i}^P$ where the i stands for the i-th filter, x refers to the horizontal component and S to the extrapolation with the S-wave velocity model and analogously for the vertical component, z, downward continued with the P-wave velocity model.

To overcome the destructive interference at the source depth we compute the 2D deconvolution of the inverse extrapolated wavefields by the filters. The filters that more closely match the location and source mechanism of the extrapolated wavefields yield a stronger focusing. Thus, from the computed deconvolved wavefields $d_{x,dec}^S$ and $d_{z,dec}^P$ the maximum value, maxP and maxS, over all extrapolated depths is determined. From that we retrieve the estimated source locations, $\mathbf{x}_P$ and $\mathbf{x}_S$, the source origin times and an estimate of the source mechanism.

It is desired that the estimated source locations $\mathbf{x}_P$ and $\mathbf{x}_S$ as well as their distances to the filter location, $\mathbf{x}_K$, applied to obtain the locations are close from

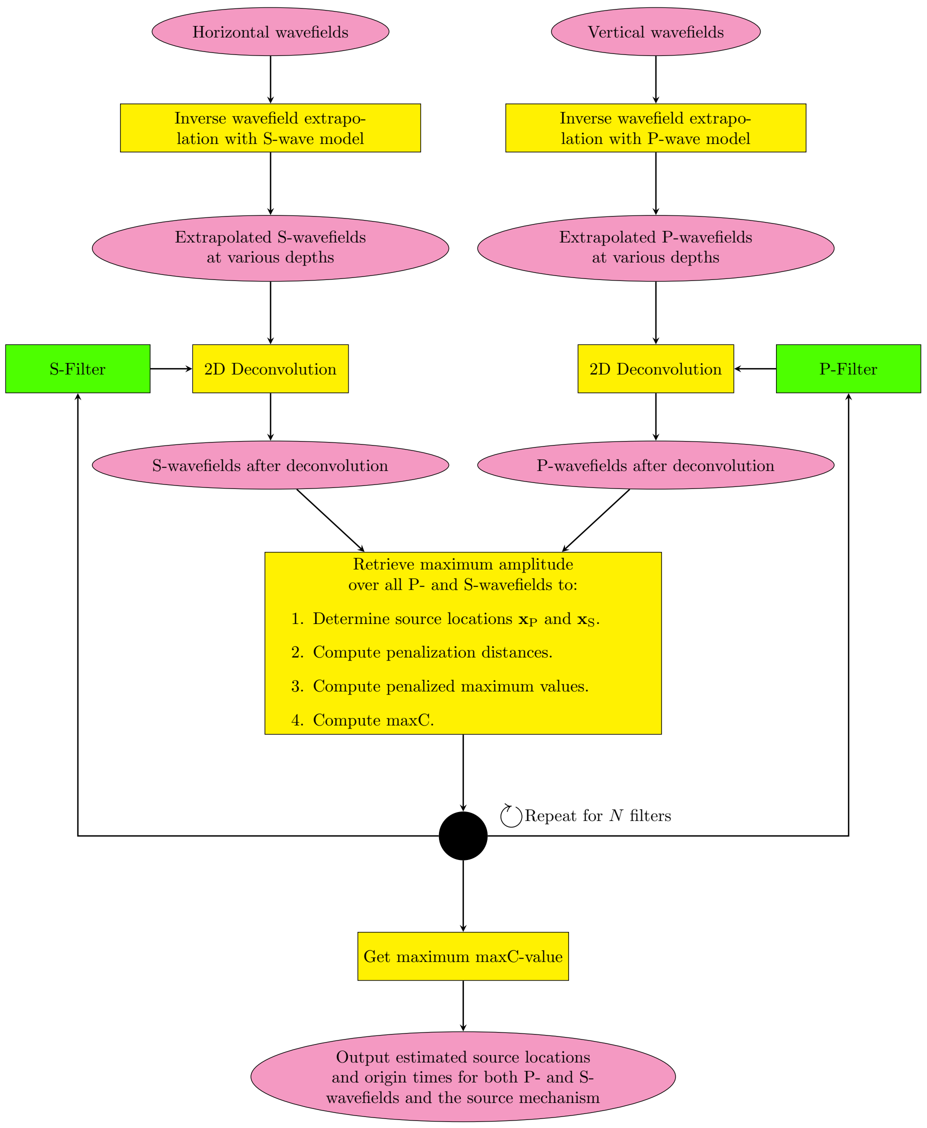

Horizontal wavefields
Vertical wavefields
Inverse wavefield extrapolation with S-wave model
Inverse wavefield extrapolation with P-wave model
Extrapolated S-wavefields at various depths
Extrapolated P-wavefields at various depths
S-Filter
2D Deconvolution
2D Deconvolution
P-Filter
S-wavefields after deconvolution
P-wavefields after deconvolution
Retrieve maximum amplitude over all P- and S-wavefields to:
1. Determine source locations $\mathbf{x}_\mathrm{P}$ and $\mathbf{x}_\mathrm{S}$.
2. Compute penalization distances.
3. Compute penalized maximum values.
4. Compute maxC.
Repeat for N filters
Get maximum maxC-value
Output estimated source locations and origin times for both P- and S-wavefields and the source mechanism

Figure 2.2: *Procedure to retrieve source location, origin time and mechanism.*

each other. Therefore, we compute penalization distances d_P and d_S as,

$$d_P = \|\mathbf{x}_P - \mathbf{x}_S\| + \|\mathbf{x}_P - \mathbf{x}_K\| \tag{2.2.5a}$$

$$d_S = \|\mathbf{x}_P - \mathbf{x}_S\| + \|\mathbf{x}_S - \mathbf{x}_K\|, \tag{2.2.5b}$$

which are used to penalize estimated source locations that are far from each other and their respective filter location, by computing weighted maximum values $\mathrm{maxS_w}$ and $\mathrm{maxP_w}$:

$$\mathrm{maxP_w} = \frac{\mathrm{maxP}}{d_p} \tag{2.2.6a}$$

$$\mathrm{maxS_w} = \frac{\mathrm{maxS}}{d_s}. \tag{2.2.6b}$$

Finally, the sum of the penalized maximum values, maxC, is computed:

$$\mathrm{maxC} = \mathrm{maxP_w} + \mathrm{maxS_w} \tag{2.2.7}$$

These calculations are performed for all filters and once the maxC-value for all filters has been computed, the maximum over all maxC-values is determined. This yields the estimated locations for both P- and S-wavefields, the source origin time and the estimate of the source mechanism.

2.3 Results

■ 2.3.1 Layered 2-D medium

To introduce the method, we start with a simple four-layered elastic model as shown in Fig. 2.3. This figure shows the locations of the filters and sources. Earthquakes usually rupture along faults which can be modelled as double-couple (DC) sources, representing a shear dislocation. The filter database consists of DC sources with inclinations 0°, 15°, 30°, 45°, 60° and 75° and are located at the positions of the black squares in Fig. 2.3. The filter locations are only placed in the center of the model in order to investigate whether retrieving locations for sources laterally further away from the model can still be accurately recovered. Since in this example we have a layered model it may not be strictly necessary to have filters at many lateral positions because of the lateral invariance. However, this is possible only to a certain extent, as the focusing is not only dependent on the source mechanism and the model but also on the recording aperture. For computational reasons we only used a few inclinations and filter locations. With this database we try to recover the locations and source mechanisms for the numbered sources positioned at the white squares in Fig. 2.3.

Receivers are spanned along the surface from 20 to 4980 m with 20 m intervals. The synthetic surface data is generated with the elastic solver Salvus [*Afanasiev et al.*, 2019a]. We inverse extrapolate the vertical-component data with the P-wave velocity model (d_z^P) and the horizontal-component data with the S-wave velocity model (d_x^S). An example wavefield as well as two examples of filters designed for the

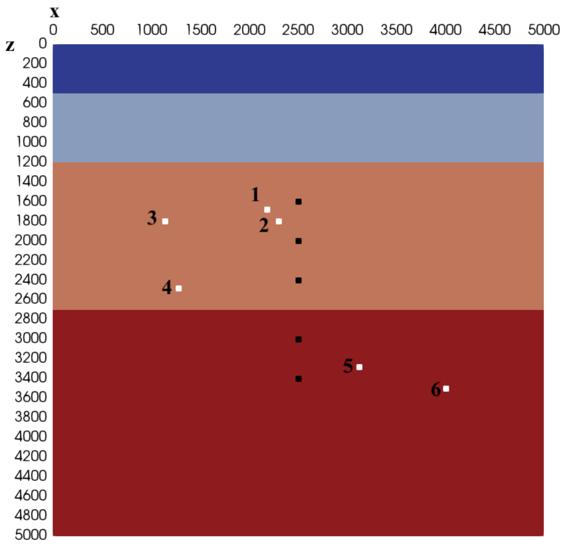


Figure 2.3: *2D model: P-wave velocities from top to bottom: 1500, 2200, 3000 and 3500 m/s and S-wave velocities from top to bottom: 600, 930, 1200 and 1400 m/s. Filter locations (black dots) and source locations (white dots).*

Table 2.1: *Summary of results for the four layered model: Source locations and double-couple inclination (θ_s) and their estimated locations as well as the filter with source mechanism information that yielded the best result.*

Source No.	(x, z)-location (m)	θ_s (°)	estimated (x, z)-location (m)	Filter θ_s°; depth (m))
1	(2180, 1680)	75	(2180, 1700)	75; 1600
2	(2300, 1800)	67	(2300, 1800)	75; 2000
3	(1140, 1800)	67	(1160, 1800)	60; 1600
4	(1280, 2480)	25	(1300, 2450)	30; 3000
5	(3120, 3280)	25	(3120, 3300)	30; 3000
6	(4000, 3500)	38	(3990, 3550)	30; 3000

four-layered model are shown in Fig. 2.4. The data are inverse extrapolated in the depth interval 1000 to 4500 m with steps of 100 m. In this example we retrieve the source location and mechanism based solely on the maximum absolute amplitude of the inverse extrapolated and deconvolved wavefields.

For source number 1 located at (x=2180, z=1680) m with a DC inclination of 75°, the location is accurately determined at (x=2180, z=1700) m with the filter at depth 1600 with θ_s=75°, thus also estimating the source mechanism correctly, as summarized in Table, 4.1.

The location of source number 2 is also accurately recovered, see table 4.1. Due to the coarse filter sampling in both space and source mechanism the estimated θ_s

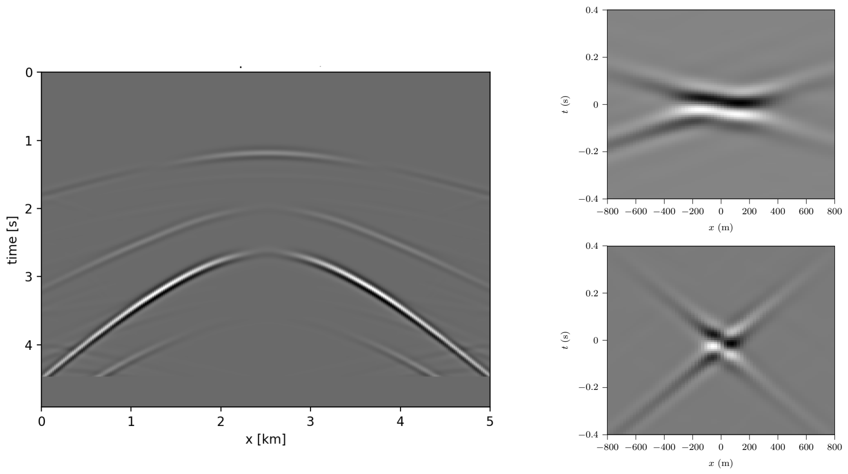


Figure 2.4: *Vertical particle velocity data for a source with $\theta_s = 45°$ in 4-layered medium (left) and two filter examples h_z^P (top right) and h_z^S (bottom right).*

does not exactly match the θ_s of 67°. However, the estimated source mechanism is close with θ_s=75°.

The locations and source mechanisms recovered for source number 3, which is horizontally positioned much further away from the filters, are also accurately estimated, see table 4.1. Source number 4 is also horizontally further away from the filters. The estimated source mechanism is also close to the actual source mechanism. However, while the maximum absolute value for both the inverse extrapolated and deconvolved P- and S-waves is at the same horizontal location, the depths are different. For the P-wave the depth is estimated at 2400 m while for the S-wave the depth is found at 2500 m. Therefore, the reported depth in table 4.1 is reported at 2450 m. Finally, the recovered location and mechanism for source number 5 are accurate, whereas for source number 6 the location is slightly worse but still good. This is expected as source number 6 is located furthest away from the filters compared to the other sources.

Finally, the origin times for all sources were found at less than 0.04 seconds from the true origin time.

■ 2.3.2 Complex 2-D medium: Noise-free data

Next, we apply the method to a single event modelled using a structurally complex elastic 2D model, resembling the geology close to Annerveen, a small town in the North of the Netherlands, lying above a gas field. The model (Fig. 2.5) is based on seismic data and includes a salt layer that is located just above the gas-producing field. Receivers are spanned along the surface in 20 m intervals ranging from 20 to 8980 m. The location of the point source is denoted by the large black square at (x=5900 m, z=3700 m) and it is defined as a double-couple source with an

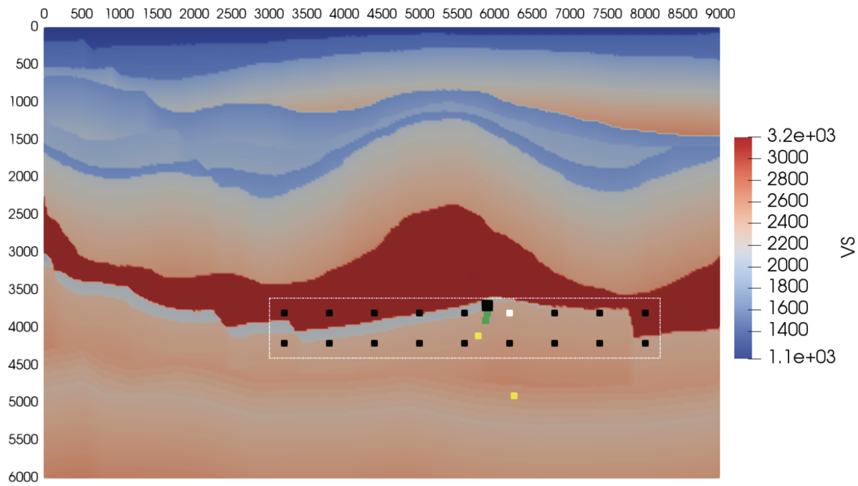


Figure 2.5: *Annerveen S-wave velocity model. Source location (large black square), filter locations (small black squares), best filter location (white square), estimated source locations without deconvolution (yellow squares) and estimated source locations with deconvolution (green squares). White-dotted rectangle denotes area for designing filters.*

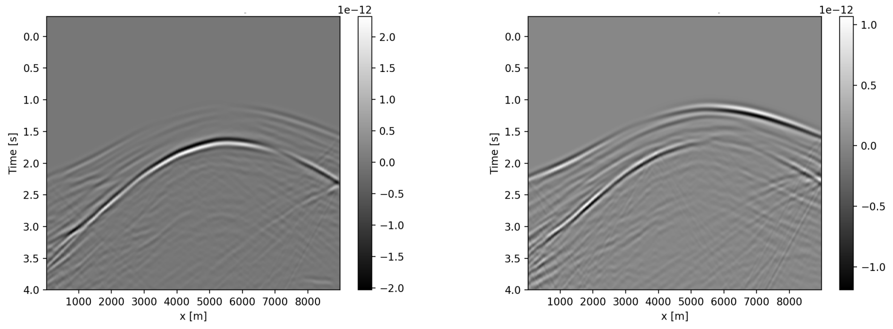


Figure 2.6: *Horizontal (left) and vertical (right) wavefields from the double-couple point source ($\theta_s = 48^\circ$) in the Annerveen model, located at the large black dot in v_x (left) and v_z (right) recording input for extrapolation with the S-wave and P-wave velocity model, respectively.*

inclination of 48°. The horizontal and vertical component of the particle-velocity wavefields are shown in Fig. 2.6. As in the example above we seek the absolute maximum value in the inverse extrapolated and deconvolved wavefields to determine the source parameters.

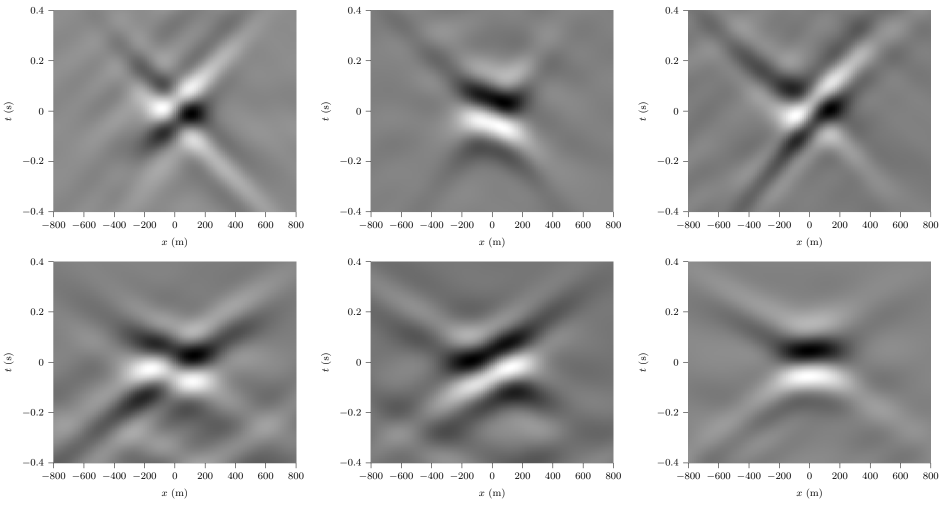


Figure 2.7: *Filter examples with different (x,y) locations and DC inclinations: (3800, 3800) m, (7400, 3800) m and (5400, 3800) m and DC-inclinations (60, 15 and 0) from left to right. Top row: S-wave filters. Bottom row: P-wave filters.*

The white dotted rectangle in Fig. 2.5 delineates the area used to design the filters denoted by small black squares and one white square. At each of those filter locations, seven different double-couple sources (DC) are considered. The DC sources have different angles, θ_s, from the horizontal, ranging from 0° to 90° with increments of 15°. With there being 18 filter locations, this results in a total of 126 filters, N_{filt}, for each H_x^S and H_Z^P. Some of the P- and S-filters are shown in Fig. 2.7.

The surface wavefields are inverse extrapolated to various depths between 1500 and 5000 m in 100 m intervals. As described above, the horizontal wavefield is extrapolated with the S-wave velocity model and the vertical wavefield with the P-velocity model. A few inverse extrapolated wavefield snapshots at depths of 2000 m, 3700 m and 5000 m of the horizontal surface data (displayed in Fig. 2.6) are shown in Fig. 2.8. From left to right we have the wavefields at a depth above, at and below the true source depth, respectively.

First, we attempt to retrieve the source location by directly seeking the maximum amplitude of the inverse extrapolated wavefields without deconvolution. The retrieved locations of the two extrapolation components are denoted by the yellow squares in Fig. 2.5. The result is not very satisfying, especially with respect to the source depth and has to do with destructive interference.

Next, we try to improve the source parameter estimates by 2D deconvolution. Applying the procedure described above, i.e. computing the deconvolution of each inverse extrapolated wavefield by all filters in the database and finding the optimal

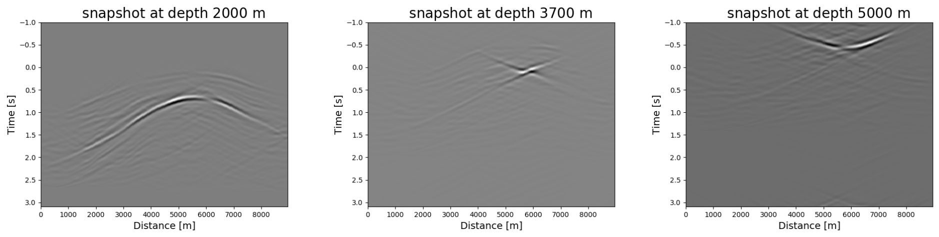


Figure 2.8: *Horizontal wavefields extrapolated at three different depths: 2000 m, 3700 m and 5000 m from left to right. The source depth is at 3700 m.*

solution, the best estimate is found using the filter located at the white square with DC inclination of 45°. The retrieved locations for the two extrapolation components after deconvolution by that filter are much closer to the location of the source (green squares in Fig. 2.5) compared to the locations retrieved without deconvolution (yellow squares in Fig. 2.5). The estimated source mechanism is also close to the actual source mechanism.

2.3.3 Complex 2-D medium: Noisy data

In this example we add field noise to the synthetic data shown in Fig. 2.6. The field noise added to the synthetics was recorded in Groningen by 92 multi-component receivers. Since the synthetic data have more traces than the noise, the noise is linearly interpolated to 449 traces, equaling the number of traces in the synthetics.

Experimental setup

To take into account velocity model uncertainties, the Annerveen model used to generate the synthetic surface wavefields is slightly modified. To generate the filters and for inverse wavefield extrapolation the same Annerveen model as before is used. The differences are present in the salt layer and the layer above the salt contoured by white lines in Fig. 2.9. The contoured layer is 200 m/s higher in the S-wave velocity and 100 m/s higher in the P-wave velocity compared to the model used for the filters. The salt layer is constant with 3200 m/s in S-wave velocity and 4700 m/s in P-wave velocity for the filters. In the true model the salt layers shear-wave velocity varies between 3300 and 3350 m/s and between 4700 and 4750 m/s in the P-wave velocity. All forward modelling computations are carried out with Salvus [*Afanasiev et al.*, 2019b].

As in the previous example the locations denoted by black squares in Fig. 2.9 represent the locations used to design the filters and the same filters described above were used. The source location is denoted by the white square in Fig. 2.9 and has DC inclined by 48 degrees. The noise-free horizontal, d_x, and vertical, d_z, surface data are shown in Fig. 2.6. Again, the surface wavefields are inverse extrapolated

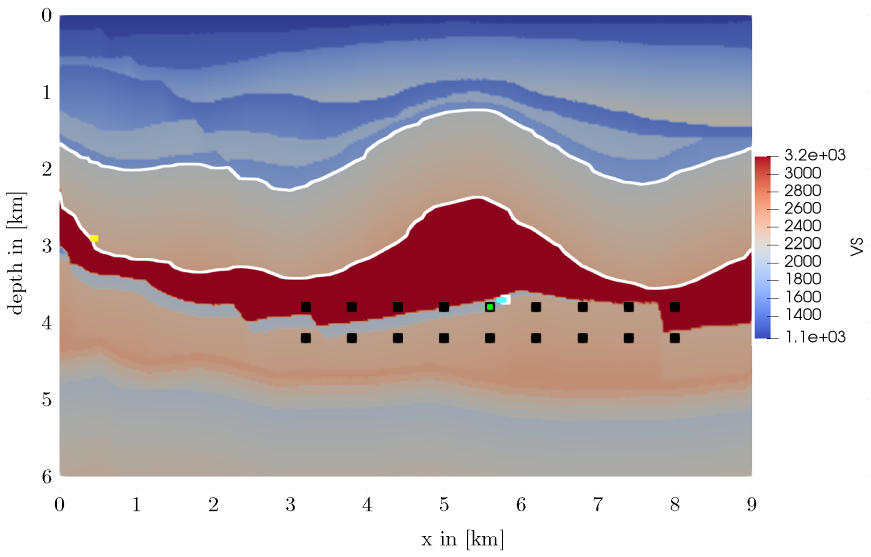


Figure 2.9: *Annerveen model. Filter locations (black dots), source location (white dot), estimated P/S locations (cyan), filter location yielding best result (green dot), locations estimated without deconvolution (yellow), salt layer (red), layer above salt denoted by white lines.*

to various depths between 1500 and 5000 m in 100 m intervals. With an interval of 100 m the computational cost is kept at an acceptable level and still provides an acceptable depth resolution for this scenario.

Retrieval of source parameters

As mentioned we add field noise to the synthetics. The surface wavefields with the added field noise are shown in Fig. 2.10. The S/N is 0.264 for the P-wave in the vertical component and 0.303 for the S-wave in the horizontal component. The surface wavefields are inverse extrapolated through the respective velocity model with the WLSQ inverse extrapolation method [*Thorbecke et al.*, 2004]. If we try to directly recover the source location based on the maximum amplitude in the inverse extrapolated wavefields without deconvolution, the retrieved source location is completely off and the maximum amplitude is found somewhere in the noise. The retrieved location is denoted by the yellow square in Fig. 2.9 at the left of the model. To the contrary, when we apply deconvolution and follow the steps summarized in Fig. 2.2 the retrieved source locations for both the P- and S-extrapolated wavefields are very close to the actual source location as denoted by the cyan colored square in Fig. 2.9. The filter used to obtain this result is located close to the source as well and is denoted by the small green square and it has a DC inclined by 45°, which

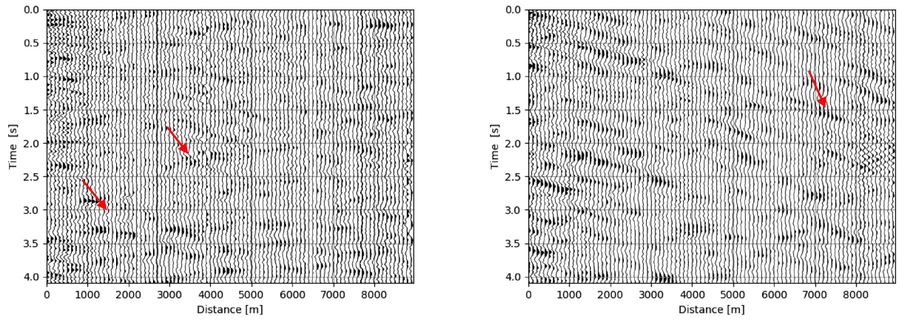


Figure 2.10: *Horizontal (d_x, left) and vertical (d_z, right) particle velocity data superimposed with real spatially correlated noise. These serve as input for the inverse wavefield extrapolation with the S-wave and P-wave velocity model, respectively. Red arrows highlight areas where the waves are visible. Every fifth trace is plotted for illustration purposes. The noise-free equivalent is shown in Fig. 2.6.*

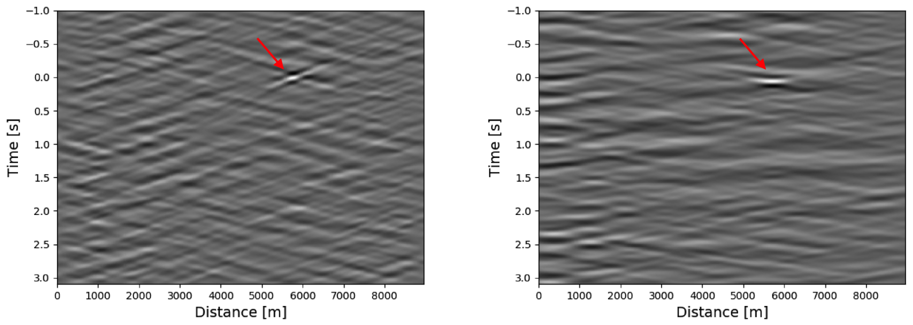


Figure 2.11: *Inverse extrapolated surface wavefields (particle velocity d_x left, d_z right) at retrieved source depth after deconvolution. Red arrows indicate focused signal position*

is close to the true DC inclined by 48°. The inverse extrapolated wavefield after deconvolution at the recovered source depth is shown in Fig. 2.11. The focused signal can be clearly observed in both the horizontal and vertical wavefields. The S/N improved significantly after inverse extrapolation and deconvolution with the optimal filter from 0.26 to 2.65 for the P- and from 0.30 to 2.24 for the S-waves.

By tracking the normalized maximum amplitudes in the downward continued wavefields at each depth level the effect of taking the source mechanism into account by deconvolution vs not taking the source mechanism into account becomes apparent as well. Fig. 2.12 shows the normalized maximum amplitudes of the wavefields over

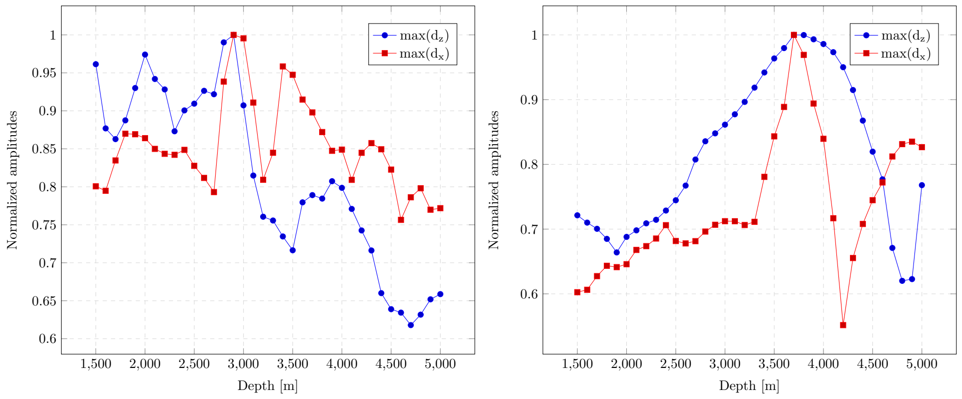


Figure 2.12: *Change of maximum absolute amplitude as function of depth. Left: maximum amplitude variations from extrapolated wavefields without deconvolution. Right: maximum amplitude variations after deconvolution with the optimal filter.*

the extrapolation depth. Without deconvolution no clear peak is visible and the plot looks noisy. On the other hand the plot showing the maximum amplitude of the downward continued wavefields after deconvolution with the optimal filter shows a clear peak that forms at the correct source depth.

2.4 Discussion

The source focusing pattern reconstructed by inverse wavefield extrapolation depends not only on the source mechanism, but also on the overburden and acquisition design. This limits the application of the method to areas with well known velocity models. However, for the situation of monitoring oil and gas fields it can be assumed that at least the P-wave velocity model is well known from active reflection surveys. Nonetheless, the accuracy of the velocity model plays a very important role for most source-localization methods and this is expected to be the case also for this method. Further studies are needed to assess how strongly velocity variations affect the localization process.

Without prior knowledge about the expected source mechanisms and hypocenters in the region of interest, a large number of filters needs to be designed. This naturally increases the computational resources to both design and run the filters, also during monitoring due to the need to compute the deconvolution of each extrapolated wavefield by deconvolving by the entire filter database. The number of filters could be reduced if any prior knowledge about the expected source locations and mechanisms can be taken into account. Another approach that may reduce the computational burden could be to start with a sub sample of the filter locations and DC components and iteratively refine the result by including more filters.

The method requires regular and dense spatial sampling, which is often not the case in current monitoring practices. Further research is needed to find a way to adapt the method to sparse and irregular acquisition geometries.

One-way wavefield extrapolator operators can not handle guided waves such as those encountered in boreholes. If many such waves would be present in the surface data these would represent noise for the one-way operators. If these would prevent the source location algorithm from recovering accurate locations, they would have to be removed in a preprocessing step.

In this chapter the vertical P-wave data was inverse extrapolated with the P-wave model and the horizontal S-wave data with the S-wave model. Generalizing the method to PS and SP waves could be achieved by designing new operators which would require knowing where the conversions happen.

2.5 Conclusions

Accurate hypocenters are obtained by combining inverse wavefield extrapolation with 2D deconvolution using filters that include source-mechanism information. Additionally, an estimate of the source mechanism is obtained. The filters need to be designed for various source mechanisms and also at various locations within the model. However, the results show that even with a sparse distribution of deconvolution filters in terms of spatial location and source mechanism good results are obtained even for a complex medium and signal levels below the noise levels. Furthermore, the results showed that it is crucial to take the source mechanism into account, especially for events below the noise level.

3

Localizing microseismic events on field data using a U-Net based convolutional neural network trained on synthetic data

"I am scared that if you make the technology work better, you help the NSA misuse it more. I'd be more worried about that than about autonomous killer robots."

Geoffrey Hinton

Abstract Hydraulic fracturing plays an important role when it comes to the extraction of resources in unconventional reservoirs. The microseismic activity arising during hydraulic fracturing operations needs to be monitored to both improve productivity and to make decisions about mitigation measures. Recently, deep learning methods have been investigated to localize earthquakes given field-data waveforms as input. For optimal results, these methods require large field data sets that cover the entire region of interest. In practice, such data sets are often scarce. To overcome this shortcoming, we propose initially to use a (large) synthetic data set with full waveforms to train a U-Net that reconstructs the source location as a 3D Gaussian distribution. As field data set for our study we use data recorded during hydraulic fracturing operations in Texas. Synthetic waveforms were modelled using a velocity model from the site that was also used for a conventional diffraction-stacking (DS) approach. To increase the U-Nets ability to localize seismic events, we augmented

Published as: N. A. Vinard, G. G. Drijkoningen and D. J. Verschuur (2022), Localizing microseismic events on field data using a U-Net based convolutional neural network trained on synthetic data, Geophysics, 87 (2), doi.org/10.1190/geo2020-0868.1
Note that minor changes have been introduced to make the text consistent with the other chapters.

the synthetic data with different techniques, including the addition of field noise. We select the best performing U-Net using 22 events that have previously been identified to be confidently localized by DS and apply that U-Net to all 1245 events. We compare our predicted locations to DS and the DS locations refined by a relative location (DSRL) method. The U-Net based locations are better constrained in depth compared to DS and the mean hypocenter difference with respect to DSRL locations is 163 m. This shows potential for the use of synthetic data to complement or replace field data for training. Furthermore, after training, the method returns the source locations in near real-time given the full waveforms, alleviating the need to pick arrival times.

3.1 Introduction

Many of our society's energy requirements are provided by oil and gas that are extracted from conventional and unconventional reservoirs in the subsurface. In certain unconventional reservoirs, hydraulic fracturing (HF) is a prerequisite step for the extraction of the resources. During HF operations fluids are injected into the subsurface at high pressures to fracture the surrounding rocks [*Li et al.*, 2020]. Naturally, this leads to seismicity that is generally weak with moment magnitudes (**M**) around 0, typically referred to as microseismicity [*Van Der Baan et al.*, 2013].

Microseismic monitoring has been around for many decades and finds applications in different industries such as the mining industry to gain insights about the rock-mass response to mining activities [*Mendecki*, 1993], the hydroelectric power industry to monitor the seismicity induced by water reservoirs [*Simpson et al.*, 1988] and the geothermal industry [*Pearson*, 1981]. However, herein we focus on HF.

Continuous microseismic monitoring in a hydraulic fracturing setting provides important information to both optimize production [*Maxwell et al.*, 2002] and to decide about mitigation measures to prevent larger events [*Kao et al.*, 2018]. Some key tasks of microseismic monitoring include event detection, source-location identification, event-magnitude evaluation and source-mechanism inversion [*Li et al.*, 2019a]. The location of events is important for several reasons. First, it helps to differentiate between events linked to the current anthropogenic activity and other types of events. Second, the event locations provide a map of the created fractures. Finally, the locations provide a starting point for understanding the source mechanism.

In some situations HF is carried out in multiple stages lasting over extended time periods. For example the operations within the Duvernay Formation in Canada were carried out over 153 stages in two time periods, the first one lasting 17 and the second one 16 days [*McKean et al.*, 2019; *Rodríguez-Pradilla and Eaton*, 2020]. In another example 19 stages each lasting about 3 hours were performed between the 28 October and 10 November 2014 in the southern Sichuan Basin, China [*Chen et al.*, 2018]. Similarly, the microseismic data used in this study, monitored HF activities over a period of 3 weeks in the Barnett Shale in Texas, USA, using a near-surface permanent installation consisting of vertical component geophones [*Kratz et al.*, 2012]. In such situations, where a lot of data needs to be processed, it may be desirable to have a method that retrieves the event locations in near real-time either while the operations are ongoing or in a post-processing step.

Waveform-based source-localization methods are commonly used in the field of microseismic monitoring [*Li et al.*, 2020]. One such method involves stacking the wavefields at different stations along expected travel-time functions to improve the signal-to-noise ratio (S/N) [*Anikiev et al.*, 2014; *Trojanowski and Eisner*, 2017]. Since the source location and origin time are unknown, a grid search over these variables is required to correctly stack the waveforms. Due to the source mechanism of the events, different polarities can be observed at different stations. Therefore, an additional grid search to align the polarities at the different stations can be used, which can improve detection [*Chambers et al.*, 2014; *Anikiev et al.*, 2014].

Machine learning and in particular deep-learning techniques have been widely applied in seismology for a wide number of tasks such as seismic phase detection and picking [*Ross et al.*, 2018a; *Zhu et al.*, 2019; *Zhu and Beroza*, 2019; *Zhang et al.*, 2020a], P-wave picking and first-motion polarity determination [*Ross et al.*, 2018b], microseismic monitoring in mining [*Mousavi et al.*, 2016; *Huang et al.*, 2018; *Johnson et al.*, 2021], earthquake early warning [*Jozinović et al.*, 2020; *Münchmeyer et al.*, 2021], earthquake signal detection [*Perol et al.*, 2018; *Mousavi et al.*, 2019] and earthquake magnitude estimation [*Lomax et al.*, 2019; *Mousavi and Beroza*, 2020b]. Convolutional neural networks (CNNs) have also been applied in recent years to identify the location of seismic events. *Kriegerowski et al.* [2019] formulate the source localization problem as a regression task returning a (x,y,z)-location for the event. Furthermore, they used the waveforms from multiple three-component stations as input data to their CNN. *Mousavi and Beroza* [2020a] present a Bayesian deep learning approach to identify the location of global earthquakes from single-station observations. *Van den Ende and Ampuero* [2020] propose a location algorithm that includes the information of the receiver locations using a graph-based deep learning approach. *Zhang et al.* [2020b] use a fully convolutional network (FCN) [*Long et al.*, 2015] with multiple up-sampling layers to retrieve the source location of induced events in Oklahoma. They used the FCN to return the source location as a 3D probability density function where the peak defines the source location. In all these recent works [*Kriegerowski et al.*, 2019; *Zhang et al.*, 2020b; *Mousavi and Beroza*, 2020a; *Van den Ende and Ampuero*, 2020], the models were trained and applied to field data. In some of those publications it was suggested that synthetic data could be used to augment their data sets, especially to fill existing gaps in the field data. Synthetic waveforms have been used in the past to train neural networks to invert for source parameters of natural earthquakes [*Käufl et al.*, 2016a,b]. The neural networks used in these works were trained to return the posterior probability density functions for the focal depth, longitude and latitude of the earthquakes as well as other source parameters given the seismic waveforms as input. This was achieved by using Mixture Density Networks [*Bishop*, 1994, 1995].

In this chapter we model synthetic seismic data given a reasonable velocity model of the area of interest to generate a large labeled data set. The label associated to each synthetic waveform is represented as a 3D Gaussian distribution where the peak of the distribution is defined at the source location, as proposed by *Zhang et al.* [2020b]. The data consist of seismic waveforms from multiple stations, where only the vertical component is used. This synthetic data set is used to train a modified version of a U-Net, a supervised deep learning algorithm originally developed for biomedical image segmentation [*Ronneberger et al.*, 2015]. After training we apply the U-Net to microseismic field data recorded during hydraulic-fracturing operations in Texas, USA. During training we apply different types of data augmentations including the addition of field noise. To validate the U-Net we first apply it to a smaller number of events, so-called master events, that were defined by *Alexandrov et al.* [2020]. The U-Net that performs best on these 22 master events is then applied to all 1245 events with moment magnitude ranges between 1.7 to -0.6. We compare our predicted locations to locations computed using a diffraction stacking (DS) method [*Anikiev et al.*, 2014] and DS locations that were refined in a second step using a relative location (RL) method [*Grechka et al.*, 2015]. The event locations obtained by DS and RL are much better constrained in depth, i.e. around the injection depth, compared to the DS locations [*Alexandrov et al.*, 2020].

Our results show a good localization accuracy for the events with moment magnitudes above 0.2 and potential for localizing weaker **M** events, therefore suggesting that: (1) synthetic data can be used to train CNNs to localize field data events, (2) synthetic data can be used to augment field data sets, and (3) the 3D Gaussian distribution output could be used to restrict the search space for grid-search-based source-localization methods. Additionally, since after training an output is generated within milliseconds, we believe this to be an initial step towards real-time source localization either as a post-processing step or during the actual operations in situations where these last over several weeks.

The chapter is structured as follows. First, we describe how our deep learning (DL) model is trained and discuss some of the basic operations performed in convolutional neural networks. Here we also introduce the modified U-Net architecture that we use to reconstruct the source locations in terms of 3D Gaussian distributions. In the next section we introduce the field as well as the synthetic data used to train the network. This is followed by the localization results obtained using different types of learning strategies as well as data augmentations. Additionally, we show how the locations are affected by the value of **M**. Finally, we discuss the results of previous analyses and summarize our results.

3.2 Methods

In this section we introduce U-Nets that are based on convolutional neural networks (CNNs). This is followed by the U-Net architecture used for this study. Finally, we briefly describe the training process.

■ 3.2.1 U-Nets

U-Nets have a down-sampling path (encoder) and an up-sampling path (decoder) where the number of layers in both these paths are approximately the same. Furthermore, U-Nets contain skip connections between each encoder and decoder layer, which are used to pass information from the encoder to the decoder.

The convolutional layers take the output from the previous layer as input and compute convolutions of the input with a set of filters. The coefficients of these filters are optimized during the training stage. The input can be down-sampled by writing the crosscorrelation operation with an additional parameter that determines by how many samples the filter, k, skips over the input, y, to produce the output, x. This parameter is known as stride and the operation is called strided convolution,

$$x_{i,j} = \sum_{n=1}^{K_H} \sum_{m=1}^{K_W} k_{n,m} y_{is_h+n-1, js_w+m-1}, \tag{3.2.1}$$

where s_h and s_w define the stride along the height and width of the input and K_H and K_W denote the height and width of the filter, respectively.

In the decoder the down-sampled input is gradually up-sampled using transposed convolutions. The filters used in the transposed convolutions are also optimized in the training phase.

3.2.2 Training

As mentioned in the introduction, the network returns a 3D Gaussian distribution whose peak location is defined at the source location. To optimize the weights we used the sigmoid cross-entropy loss function, similar to *Zhang et al.* [2020b]. The sigmoid cross-entropy loss, $\mathfrak{L}$ is defined as,

$$\mathfrak{L} = -\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left[g_j^{(i)}\log(\hat{g}_j^{(i)}) + (1 - g_j^{(i)})\log(1 - \hat{g}_j^{(i)})\right], \tag{3.2.2}$$

where g_j is the target value i.e. the true 3D Gaussian distribution for the input j and $\hat{g}$ is the predicted output generated by the network. We take the sum of the sigmoid cross-entropy loss over the number of training examples, N, and the number of voxels, M. Due to the large size of the training set a stochastic gradient-descent algorithm is used. With stochastic gradient-descent, in each epoch (iteration) the entire training set is processed in smaller subsets, called a *mini-batch.* In each training step, a mini-batch of N training examples is used to compute the loss and update the weights of the network. At the beginning of every epoch the entire training set is randomly shuffled.

We used the ADAM algorithm [*Kingma and Ba*, 2015] to optimize the parameters in the network and the entire optimization process was implemented in Tensorflow [*Abadi et al.*, 2015]. We used the default values as proposed by *Kingma and Ba* [2015], but adjusted the learning rate (or step size) to 0.001. Finally, N=20 was used as batch size.

3.2.3 U-Net architecture

The U-Net used in this work receives 3D tensors of seismic waveforms as input, the exact input shape being $(1024 \times 96 \times 1)$ referring to the time, seismic trace and components, respectively, the number of components being 1 here since we only have the vertical component. For each convolutional layer a specific number of filters are used, which determine the number of feature maps in that layer. In these convolutional layers the 3D tensors are described by height, width and feature maps as (height $\times$ width $\times$ feature maps). We used 32 filters in the first five convolutional layers and 64 filters in the remaining convolutional and transposed-convolutional layers. The height and width (K_H and K_W) of all filters was set to 3. In the encoder, we gradually shrink the height and width of the previous input by computing strided convolutions. With respect to our seismic input, height corresponds to time and width to seismic traces. The shapes, i.e. the height, width and number of feature maps in each layer of the network are shown in Figure 3.1. In the first layer following the seismic input the number of feature maps is 32 whereas the height and width stay the same as in the input. In the next layer this input of

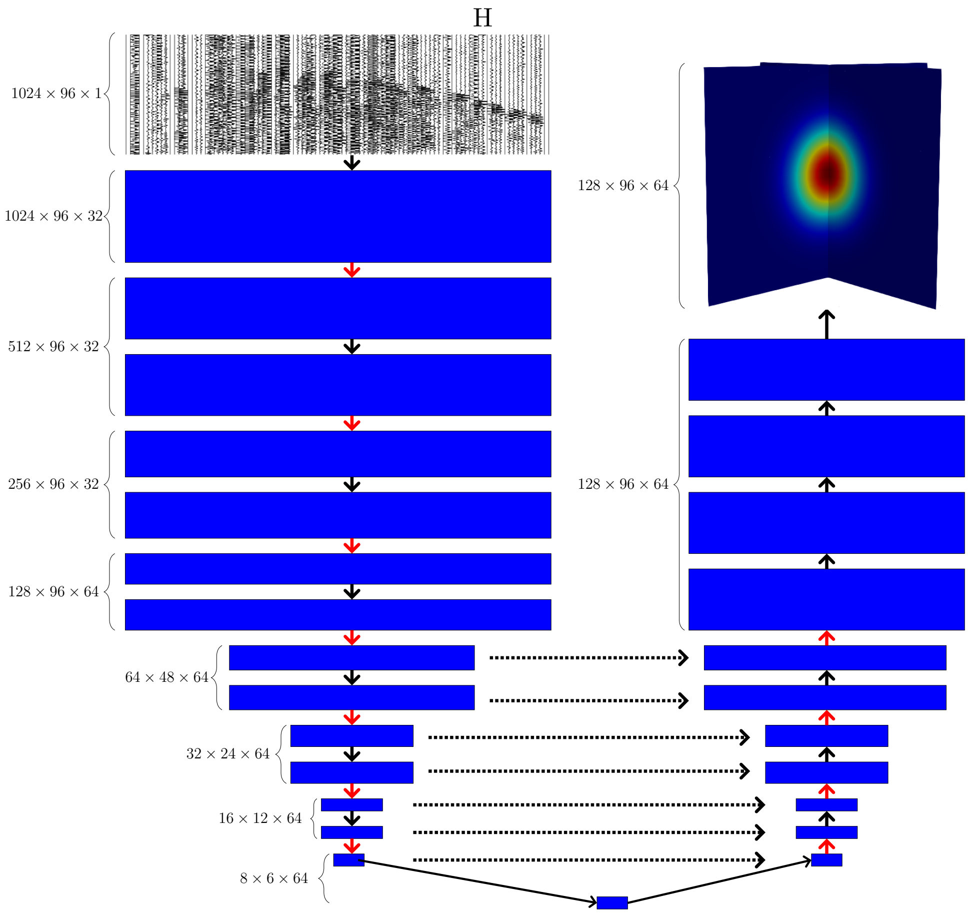


Figure 3.1: *Illustration of the U-Net architecture used throughout this work: seismic input data go through a set of convolutional and transposed convolutional layers generating a 3D Gaussian distribution as output. Black down- and up-going arrows indicate standard convolutional layers. Red down-going arrows denote strided convolutions (down-sampling) and up-going arrows denote transposed convolutional layers (up-sampling). Dashed horizontal lines indicate skip connections between encoder and decoder. The numbers next to the curly brackets indicate the dimensions of each layer.*

shape $(1024 \times 96 \times 32)$ is down-sampled using strided convolutions with strides of $(s_h = 2, s_w = 1)$ to a shape of $(512 \times 96 \times 32)$. We can see that the original input is down-sampled to a final size of $(8 \times 6 \times 64)$ before being up-sampled in the decoder to the final output size of $(128 \times 96 \times 64)$ representing a 3D Gaussian distribution.

As just described, the input and output of all convolutional layers are 3D tensors. We represent the output of the previous convolutional layer, $(l-1)$, as $\mathbf{z}^{(l-1)} \in \mathbb{R}^{Z_H \times Z_W \times C_i}$ with height Z_H, width Z_W, and feature maps C_i, where the subscript

i stands for input. Similarly, we let the output layer, l, be $\mathbf{z}^l \in \mathbb{R}^{Z_H \times Z_W \times C_o}$ with feature maps C_o, where the subscript o stands for output. Finally, the filter $\mathbf{w}^l \in \mathbb{R}^{K_H, K_W, C_o, C_i}$ has four dimensions. A single activation unit in the l-th convolutional layer can be computed as,

$$z_{h,w,o}^{(l)} = \phi \left(\sum_{i=1}^{C_i} \sum_{n=1}^{K_H} \sum_{m=1}^{K_W} \left\{ w_{n,m,i,o}^{(l)} z_{hs_h+n-1, ws_w+m-1, i}^{(l-1)} \right\} \right). \tag{3.2.3}$$

Equation 3.2.3 computes the sum of the strided convolution of the filter, $\mathbf{W}_{i,o}^{(l)}$, and the feature map of the previous layer, $\mathbf{z}_i^{(l-1)}$, and applies a nonlinear activation function $\phi(\cdot)$.

We used the (non-linear) rectified linear unit, ReLU, [*Nair and Hinton*, 2010] as the activation function in both the encoder and decoder, $\phi(x) = max(x, 0)$. We then apply batch normalization [*Ioffe and Szegedy*, 2015], which normalizes the output feature maps over the training batch by subtracting the mean and dividing by the standard deviation of the batch. In contrary to the classical U-Net architecture, which contains skip connections between each encoder and decoder layer, we only added skip connections in the deeper layers of the U-Net as represented by the horizontal arrows in Figure 3.1. The skip connections concatenate the feature maps from a layer of the encoder to the feature maps in one of the decoder layers [*Ronneberger et al.*, 2015] and speed up convergence during training [*Li et al.*, 2017]. In the last layer we used the sigmoid activation function, $S(x) = e^x/(e^x + 1)$, to map every voxel of the input, x, into a range of values between 0 and 1.

3.3 Field data and training data set

In this section we introduce the field data used to evaluate the networks that were trained with the synthetics. The field data are accompanied by an earthquake catalog and a P-wave velocity model.

■ 3.3.1 Field data

The data set contains microseismic events caused by hydraulic fracturing operations from 2010 in the Barnett Shale Formation in the Fort Worth Basin in Texas, USA [*Alexandrov et al.*, 2020]. The data were acquired by 543 vertical component geophones that were placed in shallow boreholes spanning an area of approximately 144 square kilometers. Each borehole was equipped with 3 geophones at 30, 45 and 60 meters below the surface. To limit the amount of data (in order to reduce memory and computational costs), we only kept the deepest geophones, which amounts to 181 geophones. From these, 85 receivers at large offsets had strongly attenuated signal and poor S/N. They have also been removed. This left us with 96 receivers covering the source region (Figure 3.2).

The field data set consists of 1245 detected events saved in fixed time windows of roughly 2.8 seconds each. The events were previously detected and located by *Alexandrov et al.* [2020] using a migration-type diffraction-stacking (DS) technique

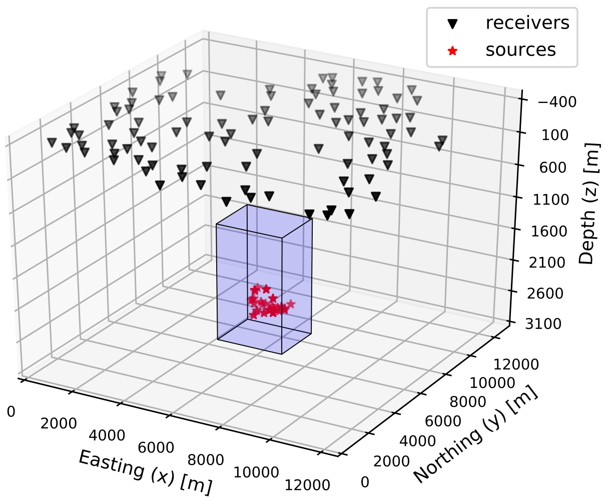


Figure 3.2: Receiver locations (black triangles), source region of interest (shaded cuboid) and source locations (red stars).

[*Anikiev et al.*, 2014] and some locations were further refined in a second step using a relative location (RL) method [*Grechka et al.*, 2015]. We will refer to this method as DSRL. The relative locations were computed with respect to 27 master events [*Alexandrov et al.*, 2020]. 22 of those 27 master events are present in our data set and those have moment magnitudes between 0.3 and 1.6. We use those 22 master events as a validation set to chose the best performing U-Net that will be used to predict the locations of all 1245 events in the field data set. Before the data enter the network we apply a band-pass filter of 5-50 Hz and normalize the data by dividing them by their maximum amplitude value. Three examples of the field data after band-pass filtering are shown in Figure 3.3.

3.3.2 Generating synthetic data

We used the reflectivity method [*Kennett and Kerry*, 1979] to generate the synthetics using the open source software ERZSOL3 [*Kennett*, 2005], because it is fast and accurate for the situation of modeling 3D data given a 1D velocity model. The reflectivity method computes the response of layered media from a point source, represented by the moment tensor. We defined the source region of interest within the 3D space shown in Figure 3.2 by the blue-shaded cuboid (it ranges from 5700 to 8300 meters in Easting, from 3700 to 5900 meters in Northing and from 1200 to 3000 meters in depth). We simulated 51200 earthquakes at random locations within the region of interest. For each event we defined a random moment tensor in terms of rake, dip and strike and therefore only consider pure double-couple sources. We

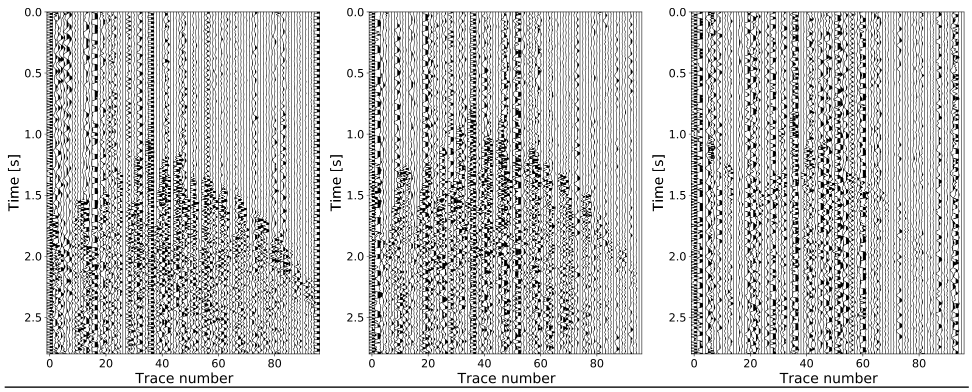


Figure 3.3: Bandpass-filtered examples of three master events with decreasing signal quality from left to right. Percentile clipping was applied to better visualize events. Shown entire waveforms are used as input to the U-Nets (without added percentile clipping).

limited the degrees for strike, dip and rake in the intervals [0, 360], [15, 85] and ±[15, 150], respectively. We checked that including a wider range of angles did not improve the quality of the results. The source center frequency was also randomly selected in the frequency range [20, 24] Hz.

3.3.3 Training data set

Supervised machine learning requires an input-output pair during the learning phase. As described above, the U-Net takes a 3D tensor of seismic waveforms as input. The output is represented as a 3D Gaussian distribution with the peak at the source location defined as,

$$g(x, y, z) = exp\left(-\left(\frac{(x-x_s)^2}{2\sigma_X^2} + \frac{(y-y_s)^2}{2\sigma_Y^2} + \frac{(z-z_s)^2}{2\sigma_Z^2}\right)\right), \tag{3.3.1}$$

where (x_s, y_s, z_s) represent the source coordinates, (x, y, z) represent all coordinates within the 3D space of interest and σ_X, σ_Y, and σ_Z represent the spread of the Gaussian distribution in each dimension. The spread of the distribution needs to be selected prior to training. There is a trade-off between the resolution and the rate of convergence during training: The smaller the spread the sparser the 3D output will be with most values being nearly zero with a few higher values around the true source location. This will lead to poor convergence as the loss function compares the voxel-wise difference between the output and the true distribution. On the contrary a large spread decreases resolution but increases convergence. We tested different spreads and found a spread of 200 m to yield best results. The 3D output space extends from 5500 to 8500 meters in Easting, from 3500 to 6100 meters in Northing and from 1000 to 3200 meters in depth. Thus, the size of a voxel is 23 m in Easting, 27 m in Northing and 34 m in depth.

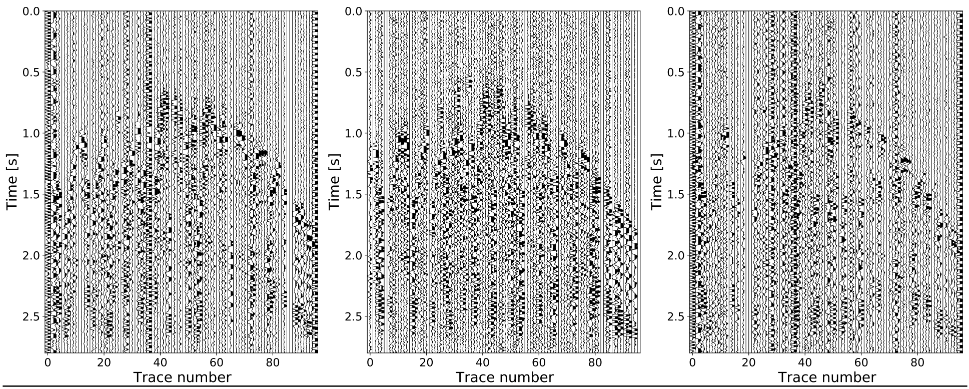


Figure 3.4: *Three synthetic events augmented with field noise as used during training.*

Data augmentations

To simulate more realistic data, we applied a few data augmentations to the synthetic data during training. In addition to fixed time windows containing detected events, we have an additional 4 hours of continuous data, from which we selected noise windows where no events were detected according to the catalog. This selected field noise can be used to augment the training set during training. Since undetected events could be present within the selected field noise windows we randomly perturb the noise windows by flipping, permutating and time-shifting the individual field-noise traces.

The following data augmentations were applied to the synthetic input data: First, we randomly bulk time-shift the data. Second, we added random band-pass filtered Gaussian noise of varying intensity in each trace. Gaussian noise does not represent field noise and is of limited use if strong field noise is present in the data. However, it is a good way to avoid zero-valued entries before and after the event in the synthetics and to artificially augment the size of the data set. Third, we add continuous field noise to the data. As a final augmentation we applied station-dropout as proposed by *Kriegerowski et al.* [2019], which means that we randomly mute between 5 and 20 traces. A few training examples with applied data augmentations are shown in Figure 3.4. After applying all data augmentations the data is normalized by its maximum absolute value before it is passed to the network.

Training and validation set and evaluation metrics

We use the 51200 synthetic waveforms and their known locations as a training set and the 22 master events as validation set. The purpose of the validation set is to ensure that the network is not overfitting to the training data and to select the best performing U-Net. To this end we evaluate the performance of the trained model on the validation set. If the discrepancy in performance between the two is high, with much better performance for the training set, the network is said to be overfitting.

We used the Dice-similarity coefficient [*Dice*, 1945] (C_{Dice}) as evaluation metric,

$$C_{Dice}(g, \hat{g}) = 2\frac{g \cap \hat{g}}{g + \hat{g}}, \tag{3.3.2}$$

where again g is the target (true 3D Gaussian distribution), $\hat{g}$ is the 3D output distribution predicted by the model and $\cap$ is the symbol for intersection. Before computing the C_{Dice} we clip all values in the target and network output distribution with values above a threshold of 0.1 to 1 and the rest to 0.

3.4 Results

In this section, we present the results of the trained U-Nets applied to the field data. We start by showing the results for the U-Net described previously, which was trained on synthetic data using all data augmentation steps described before, applied to the 22 master events selected as validation set. We will refer to this model, trained with all data augmentation steps (including field noise) and with skip connections, as U-Net A. Next, we show and discuss the results for (1) a U-Net trained without field noise, U-Net B and (2) a model trained without skip connections, i.e., an FCN. U-Net A is our baseline model, to which we compare the results of the other models.

Since the optimization method is stochastic and therefore gives slightly different results each time, we trained each deep neural network (DNN) ten times for 20 epochs. We evaluate the DNNs using the validation set by computing the C_{Dice}-coefficient between the predicted output and the desired 3D Gaussian distribution, which is constructed using the location given by the diffraction-stacking method.

■ 3.4.1 U-Net A: Baseline model

We apply the synthetic-data trained model to the 22 master events and compute the C_{Dice} value between the predicted and desired output, using the master locations computed by diffraction stacking to construct the desired output. The C_{Dice}-value over those 22 master events is 0.78, which is close to the C_{Dice}-value reached at the last epoch of training, being 0.81. To visualize the results, we take cross-sections along the horizontal and vertical planes of the 3D output at the maximum value of the Gaussian distribution. The predicted Gaussian distributions are shown in Figure 3.5. The black dots denote the DS-locations and the white stars mark the peaks of the predicted Gaussian distributions. We can observe that all 22 events are well localized within the spread of the Gaussian distribution. Larger differences are observed in depth compared to the epicenter locations, which could be due to the surface acquisition and/or differences between the synthetic and real waveforms. We estimate the hypocenter from the peak of the Gaussian distribution and compare with the hypocenter computed by DS. The results of the mean peak value of the predicted output and the mean hypocenter, epicenter and focal depth differences are summarized in Table 3.1. The mean depth, epicenter and hypocenter differences using the baseline model are 96, 82 and 135 meters, respectively.

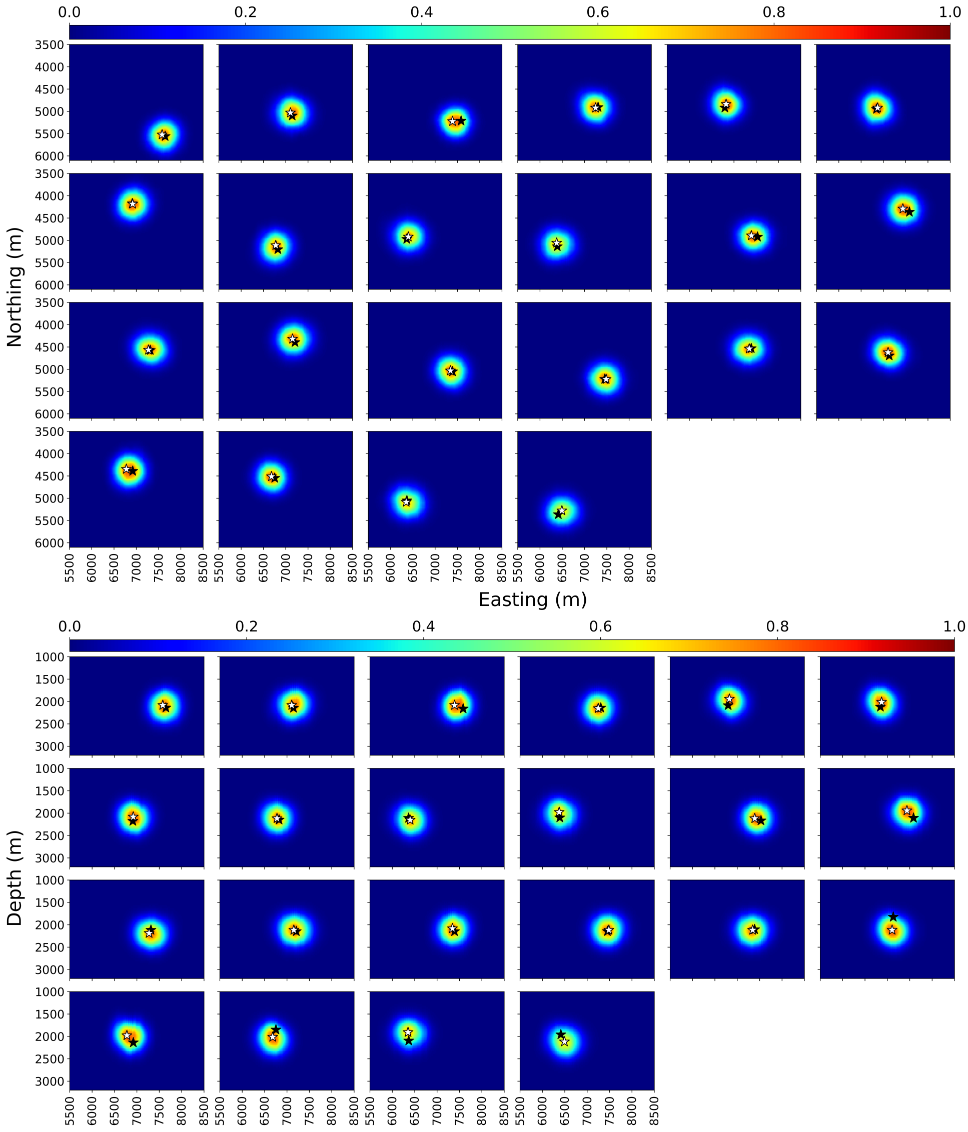


Figure 3.5: *U-Net A output cross-sections: (a) horizontal cross-section with Easting along horizontal and Northing along vertical axis and (b) vertical cross-section with Easting along horizontal and depth along vertical axis. Black stars represent DS cataloged locations and white stars are placed at highest-valued voxel in U-Net A's output. Each panel corresponds to the output of one of the 22 master events with* $\boldsymbol{M}$ *between 0.3 and 1.6.*

Table 3.1: *Mean peak value of DNNs over 22 master events and their location differences compared to DS catalog.*

Model	Peak value	Hypocenter difference (m)	Epicenter difference (m)	Focal depth difference (m)
U-Net A	0.79	135	82	96
U-Net B	0.59	302	175	223
FCN	0.75	177	54	163

3.4.2 U-Net B: no field noise

U-Net B is trained with the same U-Net architecture as U-Net A, however without the addition of field noise in the data augmentation steps. The strength of the added gaussian noise was augmented by a factor of 4 for U-Net B compared to the other U-Nets since otherwise the noise level would be too low and, therefore, the model would do poorly on most of the master events. The C_{Dice}-value on the training set is 0.90 whereas on the 22 master events it is only 0.48. The high C_{Dice}-value observed over the training set comes from the fact that the data are less complex (no added field noise) and therefore the learning process is simpler. This also explains the low C_{Dice}-value on the master events, which is due to the training set not representing the field data set well enough. The mean peak value over the master events is also low and the hypocenter and focal depth distances are significantly higher compared to U-Net A, see Table 3.1.

3.4.3 FCN: no skip connections

We trained the FCN with the same architecture as the U-Net and with all data augmentation steps but without skip connections. The C_{Dice}-value at the end of training on the training set is 0.83 whereas on the master events it is quite a bit lower, i.e. 0.70. The mean depth, epicenter and hypocenter differences are 177, 54 and 163 meters, respectively. The epicenter locations are closer to the catalog locations compared to all other models, however, the mean depth as well as hypocenter differences are larger compared to U-Net A.

3.4.4 U-Net A applied to all detected field data events

From the preceding results, we see that the peak value can change depending on the input data. For the model trained without field noise, some of the predictions had peak values around 0.1, and the predicted output did not resemble a 3D Gaussian distribution. Thus, the peak value could be used as an indicator of how reliably the model recognizes waveforms in the input data. Since we train the deep learning algorithm on synthetic data we do expect more difficulties for the models to recognize and therefore accurately predict locations of lower magnitude events with lower S/Ns. A threshold acting on the peak value could be set to only consider predictions passing the threshold. We apply U-Net A to all 1245 field data events and consider the location at the peak value of the output as the predicted hypocenter location and compute its distance from the hypocenter location given in the catalog that is

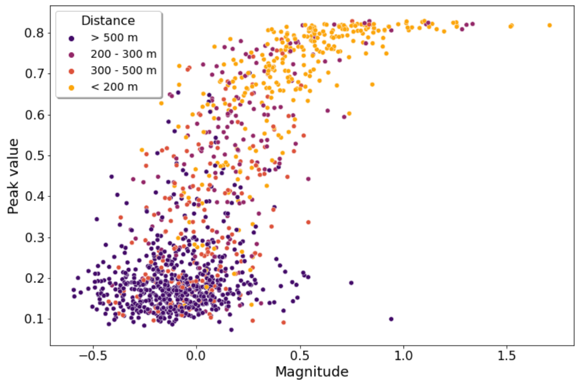


Figure 3.6: *Peak value of U-Net A output vs Magnitude of 1245 field data events. Each data point is colored according to its distance to the DSRL cataloged location.*

based on DS and further refined using the master events with the RL method. In Fig. 3.6 the peak value with respect to magnitude of all 1245 events is plotted. We observe a trend between the peak value returned by U-Net A and the **M**. Each data point is colored according to its distance to the cataloged location (based on DSRL). Most events predicted at distances smaller than 200 meters show higher peak values and also have higher magnitudes. A majority of the events with large distances between the catalog and predicted locations have peak values below 0.3 and have moment magnitudes between -0.6 to 0.3. We thus decide to set the threshold at 0.4.

We compare the predicted locations passing the threshold to the locations determined by the diffraction stacking method (DS) [*Anikiev et al.*, 2014], catalog DS, and to the locations determined by the diffraction stacking and relative-location method (RL), catalog DSRL. The main difference between the locations in catalog DS and catalog DSRL are the depth locations, which in the latter are much more concentrated along the injection depth level. For a detailed comparison between the two catalogs applied to the Texas data set we refer to [*Alexandrov et al.*, 2020]. To view the differences in locations between the catalogs and the predicted locations we draw lines connecting the predicted to both cataloged locations of each event (see Fig. 3.7). While some predicted locations still show larger differences to the cataloged locations most have a close match. A grid pattern is observed in the locations returned by U-Net A, which is due to the discretized 3D output space. We note that depth locations given by DSRL are concentrated around 2100 m depth, whereas the focal depths given by DS as well as U-Net A are scattered around 2100 m depth. Comparing the depth distribution of the located events in a histogram (Fig. 3.8) reveals that the depth distribution predicted by U-Net A more closely follows the trend of the DSRL catalog compared to the DS catalog, with most events at depths between 2100 and 2200 meters, which is the injection depth level. This could in-

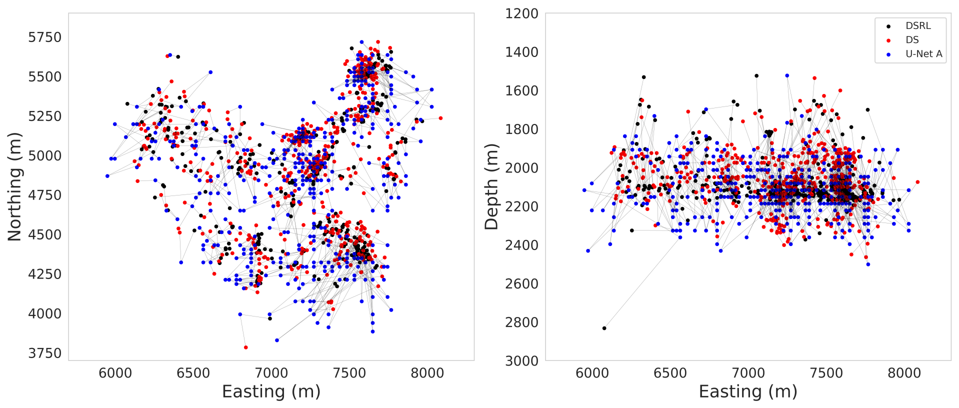


Figure 3.7: *Locations given in DS catalog (red), DSRL catalog (black) and locations predicted by U-Net A passing an amplitude threshold of 0.4 (blue). Lines connect predicted to cataloged locations.*

dicate that U-Net A is doing better at predicting the depth locations compared to DS for the events passing the threshold. The mean hypocenter, epicenter and depth distances of the locations predicted by U-Net A compared to the DSRL catalog are 214, 147 and 130 m, respectively for the 467 events that passed the threshold. The moment magnitude of those events range between -0.4 and 1.7.

With the threshold of 0.4 there are still some events that have large location differences compared to the catalogs. To focus on the predicted locations that more closely match the DSRL cataloged locations we set the threshold to 0.6 (as can be seen in Fig. 3.6). Comparing these predicted event locations to both cataloged locations, we observe a good match in epicenter locations (Fig. 3.9). Furthermore, the predicted depth locations are more concentrated around the expected depth level compared to DS. A total of 314 events pass that threshold and the mean hypocenter, epicenter and depth distances compared to the DSRL catalog are 163, 110 and 99 m, respectively, with moment magnitudes in the range -0.2 to 1.7.

Since the locations from the catalog are captured inside the 3D Gaussian distribution returned by the U-Net, the U-Net's locations could be used as initial source locations that can then be further improved with the use of other microseismic source localization methods such as RL.

3.5 Discussion

We did not address the issue of the *detection* of events in this study since we do not have continuous field data. For a practical application we would suggest to separate the detection and localization problem. The detection could for instance be made by diffraction stacking [*Anikiev et al.*, 2014; *Staněk et al.*, 2015]. Machine learning methods capable of differentiating between signal and noise have also already been

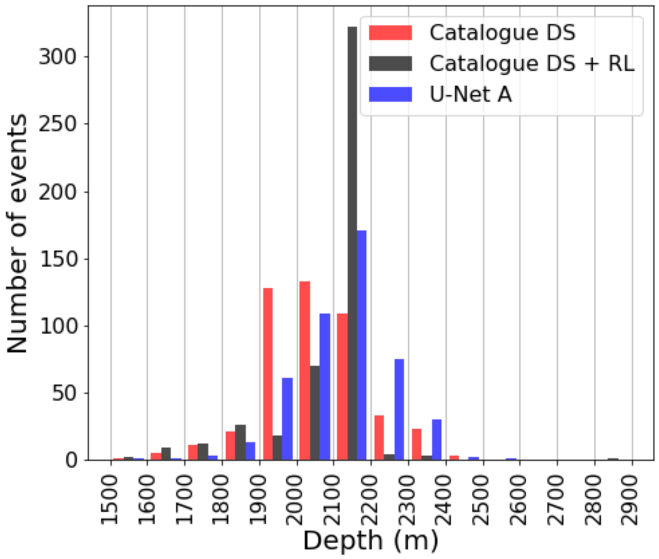


Figure 3.8: *Depth distributions of events in DS catalog (red), DSRL catalog (black) and locations predicted by U-Net A passing an amplitude threshold of 0.4 (blue).*

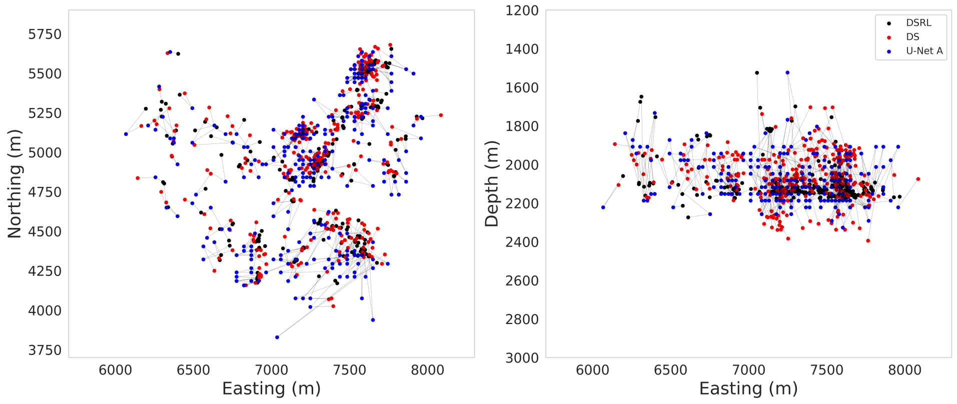


Figure 3.9: *Locations given in DS catalog (red), DSRL catalog (black) and locations predicted by U-Net passing threshold of 0.6 (blue). Lines connect predicted to both cataloged locations.*

proposed and successfully applied [*Perol et al.*, 2018; *Li et al.*, 2018; *Wu et al.*, 2018b; *Mousavi et al.*, 2019] and could also be used in combination with an event-localization method. If an event is detected, the information can be passed to the source-localization network. Alternatively, we could explore whether the U-Net trained in this study could be used as a detection method, based on the peak value returned by the model. In continuous mode a possible criterion for a detection could be that the peak value should pass a predefined threshold for a number of consecutive time windows if an event is present.

One main limitation of the method is the ability to mimic realistic field noise that can be added to the synthetics - especially to target low-S/N events. Each station in the field is subjected to local noise. Thus, if longer passive noise recordings were available for each station, these could easily be added to the corresponding traces in the synthetics during training. The network would then be able to learn directly with the noise specific to each station and possibly be able to detect and predict the hypocenter locations of lower S/N-events. This subject falls outside the scope of this work.

Contrary to previous works on earthquake source localizations using convolutional neural networks [*Kriegerowski et al.*, 2019; *Zhang et al.*, 2020b; *Mousavi and Beroza*, 2020a; *Van den Ende and Ampuero*, 2020] this work only made use of the vertical component of the seismic wavefield, because this was the only component recorded in the field. It would be interesting to study the differences of a network trained with both the vertical and horizontal wavefields and a network trained only with the vertical component. We can assume that the network with all three components would perform better since many source-localization methods make use of both P- and S-waves.

In this study, a subset of all available stations was used to reduce the amount of data needed to store the synthetic data and reduce the data input-output bottlenecks and memory footprint during the training phase of the network. However, the earthquake catalog with the source locations is based on all the available stations. Even though less stations were used the network still returned accurate source locations.

In this study we did not address the problem of localizing multiple events present in a single time window and only considered the situation where a single event is present, as this was the case in the field data. To address this issue we would suggest training a network with input data containing a random number of events, K, and similarly the target would therefore contain K Gaussian distributions. In that way the network could possibly learn to recognize whether more than a single event is present in the input and return source locations for those events.

Generating the synthetic data set for 51200 sources took 7 days running 100 jobs in parallel on 2.3 GHz Intel Xeon central processing units. Training for 20 epochs took 11 hours on a NVIDIA GeForce GTX 1080 Ti graphics processing unit. Prediction of a single event takes 0.28 seconds on a 3.1 GHz Dual-Core Intel Core i5 processor.

In this work we extended upon previous works using CNNs [*Perol et al.*, 2018; *Kriegerowski et al.*, 2019; *Zhang et al.*, 2020b] and specifically addressed the problem of missing events in the training set by generating synthetics everywhere in the model space. We wanted to test the possibility of training a network solely with synthetic data that can directly be applied to field data. This work shows that this is indeed possible, however, the ability of localizing weaker magnitude field data events decreases rapidly. It could be investigated whether augmenting the synthetics using longer recordings of passive noise from the area under investigation could lead to better localizations of weaker events or whether the weights of the network trained with the synthetic data set can be fine-tuned using field data in order to localize weaker magnitude events.

In this study we applied the method to single-component data from a hydraulic fracturing site, however, it can be applied to larger areas by up-scaling the entire experimental setup.

3.6 Conclusion

In this chapter we showed that synthetic data can be used to train a U-Net to accurately localize microseismic field data. Furthermore, we showed that augmenting the synthetic data with field noise further increases the U-Net's accuracy to localize events. After the network is trained, this method returns the source location within less than one second given the event waveforms as input. Furthermore, the retrieved locations are comparable to state-of-the-art localization methods such as diffraction-stacking and refined diffraction-stacking locations using a relative location method. In terms of depth locations the deep learning model seems to outperform diffraction-stacking as the depths are better constrained around the expected depth level for the events predicted with peak values above a set threshold. The proposed method provides locations based on the full waveform without the need for any picking, while its accuracy was better than that of a conventional diffraction stacking approach.

4

Localizing weak microseismic events using transfer learning with a deep neural network

"So easy, when you know how."

Fats Waller

Abstract Retrieving accurate microseimsic source locations induced by hydraulic-fracturing operations is an important step to gain insights about the hydraulically stimulated reservoir volume. Recently, deep neural networks (DNNs) have been proposed, which directly recover source locations from the seismic waveforms. Optimal performance of the proposed DNNs usually require large training sets. The need for a large training set can be circumvented if a previously trained DNN can be used to start the training process with its weights instead of randomly initialized weights. Those weights can then be fine tuned using a smaller training set, which is also known as transfer learning (TL). In this work we implement a TL workflow to update the weights of a DNN that was initially trained on a large synthetic data set to localize microseismic events. We present two methods of processing, namely one post-monitoring mode and one continuous mode where the processing takes place during the monitoring period. We apply the methods on field data from a hydraulic fracturing site in Texas, USA. In the first scenario a subset of the field data from the entire monitoring period is used to update the weights of the DNN, which is next applied to the remaining data resulting in mean and median distances of 227 m

Published as: N. A. Vinard, G. G. Drijkoningen, D. J. Verschuur, D. Alexandrov and L. Eisner (2022), Localizing weak microseismic events using transfer learning with a deep neural network, Geophysical Prospecting, doi: 10.1111/1365-2478.13238
Note that minor changes have been introduced to make the text consistent with the other chapters.

and 182 m, respectively, compared to the results of a good localization method. In the second scenario the DNN is updated daily with previously detected and located events and applied to the events detected the following day. Since the observed data used for training generally does not cover a wide range of source locations we enrich the training set with synthetic data. The addition of synthetics for TL ensures that the updated DNN provides accurate source locations for events with locations far from locations used during TL. TL combining synthetic and real data performs significantly better (more consistent) locations than TL without synthetics.

4.1 Introduction

The increasing demand for underground-related energy resources, such as geothermal and unconventional oil and gas reservoirs, as well as the growing interest in CO_2 sequestration and hydrogen storage requires reliable and fast methods to monitor the seismic activity around the reservoir to both optimize the underlying task and mitigate risks associated to induced earthquakes [*Gaucher et al.*, 2015; *Li et al.*, 2020]. Most of the seismicity associated to these activities is weak in moment magnitudes, $\mathbf{M}$, i.e. around and below zero [*Van Der Baan et al.*, 2013], and are called microseismic events as they are not felt at the surface. Therefore, the signal-to-noise ratio (S/N) is also poor, especially when detected by sensors close to the surface [*Li et al.*, 2019a]. Microseismic monitoring systems are set in place to detect, localize and estimate the source mechanisms and magnitudes of the induced events. The hypocenter locations provide information about the hydraulically stimulated reservoir volume or they can identify pre-existing fault systems.

In recent years several machine learning and deep-learning (DL) approaches have been proposed to identify hypocenter locations. One area of applications is focused on using machine learning and DL-based picking algorithms that are able to pick arrival times nearly as good or even better than an analyst at a fraction of the time required for manual picking [*Ross et al.*, 2018b,a; *Zhou et al.*, 2019; *Zhu and Beroza*, 2019; *Zhu et al.*, 2019; *Zhang et al.*, 2020a; *Ma et al.*, 2020]. *Ross et al.* [2018b] trained a convolutional neural network (CNN) on earthquakes with labeled P-wave picks and first-motion polarities to first detect the onset of the P-wave and next determine the polarity of the P-wave. *Ross et al.* [2018a] trained a CNN on millions of 3-component hand-labeled seismic records, that were split into records only containing P-waves, S-waves and noise, respectively. The CNN was trained to classify the input as a P-wave, S-wave or noise. *Zhou et al.* [2019] use a CNN to first detect earthquakes and next pass the detected waveforms to a recurrent neural network to pick P- and S-wave arrivals. PhaseNet [*Zhu and Beroza*, 2019] is a modified U-Net architecture [*Ronneberger et al.*, 2015] applying 1D convolutions over 3-component seismic waveforms that returns probabilities around the P-wave and S-wave arrivals and noise. *Zhu et al.* [2019] develop a CNN that can be trained on smaller training sets compared to previous works that can be applied for P- and S-wave picking of aftershocks. *Zhang et al.* [2020a] trained a CNN to classify waveforms and arrival time picking for microseismic data. To train the CNN they first convert the signal into the time-frequency domain using the continuous wavelet transform. *Ma et al.* [2020] propose a U-Net architecture for P- and S-wave classification on microseismic 3-component data. First, the data are preprocessed such that the waveforms show clearer arrival times. Next, waveforms are converted to gray-scale images and fed to the U-Net to pick the S- and P-phase arrivals.

Other DL algorithms directly return the source locations without picking of wave arrivals and directly relate observed waveforms to locations [*Kriegerowski et al.*, 2019; *Zhang et al.*, 2020b; *Van den Ende and Ampuero*, 2020; *Mousavi and Beroza*, 2020a]. *Kriegerowski et al.* [2019] accomplished this by training a CNN taking 3-component seismic waveforms from several stations as input and outputting the source locations in terms of their (x,y,z)-coordinates. *Zhang et al.* [2020b] train a deep neural network with 3-component waveforms from multiple stations as input that returns the source locations in terms of a 3D probability density function. *Van den Ende and Ampuero* [2020] propose the use of a graph neural networks, which incorporates the spatial information of the seismic stations in addition to the seismic waveforms to determine the location of the earthquakes as well as their magnitudes. *Mousavi and Beroza* [2020a] train Bayesian neural networks to estimate the location of earthquakes from single-stations.

A considerable drawback of DL methods is that large training data sets, which sample the model space well, are usually required to reach good performances. This limitation can be overcome by using a previously trained deep neural network (DNN) and refining it using a much smaller data set. This is known as transfer learning (TL) and is based on the idea that DNNs applied to similar tasks share common features [*Pan and Yang*, 2009]. In the field of geophysics TL has been applied to a variety of different problems. *El Zini et al.* [2019] used TL to detect bright spots in seismic data by first pre-training a CNN on unlabeled seismic data (unlabeled meaning that the information whether a bright spot is or is not present in the input is missing) and then fine-tuning the network on a much smaller labeled data set. By pre-training their CNN on unlabeled data they circumvent the constraint of labeled data sets required for supervised machine learning tasks. *Chai et al.* [2020] used a phase picker previously trained on 0.7 million local earthquakes (tens of km distances between sources and receivers) [*Zhu and Beroza*, 2019] and refined it to get better picks for microseismic data recorded from a meter-scale project. This was achieved using a small training data set of 3500 seismograms. In other works, TL was used by pre-training DNNs with large synthetic data sets and then fine-tuning the DNNs with field data. This has been done for the task of seismic trace interpolation using a convolutional denoising autoencoder [*Wang et al.*, 2020] and for seismic fault detection using a CNN [*Cunha et al.*, 2020].

In this work we apply TL to localize weak microseimsic events using waveforms as input. As a starting point we use a DNN that was trained with a large synthetic data set and applied to a small field-data set to retrieve the source locations of hydraulic-fracturing (HF) induced earthquakes [*Vinard et al.*, 2022]. This was achieved using a modified version of a U-Net [*Ronneberger et al.*, 2015], a type of CNN originally developed for image segmentation, which is composed of both an encoder and decoder. The encoder extracts useful features in the input (waveforms) and the decoder maps the extracted features into a 3D Gaussian distribution of location probability [*Vinard et al.*, 2022]. In the following we refer to the DNN that was trained on synthetic data as QNetSynth, where Q stands for quake and Synth for synthetic. QNetSynth reliably localizes the higher-magnitude events, however it fails to accurately localize lower-magnitude events. This is problematic for monitoring applications where the majority of the events are low in magnitude, such as observed in HF monitoring. To improve QNetSynth's performance we apply transfer learning by updating it with field data. We refer to this updated version as QNet. Furthermore, we aim for QNet to return more consistent locations compared to the diffraction stacking locations.

To train the DNN we need labeled data, meaning that we need the input (waveforms) and the known output (source locations), also called the label. This type of learning with labeled data is called supervised learning. After training, the QNet can receive new waveforms as input and return source locations.

Training a DNN with synthetic data for the task of microseismic source localization can be useful to retrieve initial locations of seismic events on real datasets. However, for low S/N events such training is insufficient. To increase the source location accuracy of the DNN trained on synthetic data, TL using field data is investigated. Furthermore, we are interested in both modes of processing: (1) post-monitoring processing when data are processed after being acquired and (2) continuous processing when the data are acquired while being processed (e.g. near-real-time or real-time processing).

In the next section we explain the transfer learning process in more detail and how the DNN is evaluated. We also discuss its application and illustrate how the methodology can be applied to a dataset from a monitored hydraulic fracturing site. The case study is investigated in both a post-monitoring processing mode as well as continuous acquisition mode. Finally, we discuss limitations and potential of this approach for future applications. Note that we do not necessarily aim at improving the quality of the locations, but its automation and consistency.

In this section we describe the transfer learning (TL) process used to predict the source locations using the synthetically trained DNN, QNetSynth. The labeled training data consists of input-output pairs, which for QNetSynth consisted of synthetic seismic waveforms as input and their corresponding source locations represented as 3D Gaussian distributions as output. The peak of the Gaussian distribution is taken at the source location, (x_s, y_s, z_s) of the event and the standard deviation, σ, has a fixed (input) value in all directions independent of the input data. The Gaussian distribution is defined as,

$$g(x,y,z) = exp\left(-\left(\frac{(x-x_s)^2}{2\sigma^2} + \frac{(y-y_s)^2}{2\sigma^2} + \frac{(z-z_s)^2}{2\sigma^2}\right)\right). \tag{4.1.1}$$

The values in the output range from 0 to 1. In supervised learning the weights of a DNN are optimized by minimizing a loss function that computes the difference between the label and the output generated by the DNN based on the current weights. After training, the DNN can be applied to new (unlabeled) data to return 3D Gaussian distributions. If the weights of the DNN can extract the relevant features from the input with ease, the returned Gaussian distribution will have a peak value of 1 at the location expected by the DNN.

The architecture of QNetSynth is shown in Fig. 4.1. QNetSynth consists of convolutional layers in the encoder where the input is gradually down-sampled as it moves further down the convolutional layers. In the decoder transposed convolutional layers gradually up-sample the previously down-sampled input. Additionally, a few skip connections are established, that pass the output from layers in the encoder to the decoder by concatenating the encoder output with the decoder output. The ADAM algorithm [*Kingma and Ba*, 2015] was used to train QNeSynth using a learning rate (step size) of 0.001, a batch size of 20 and the sigmoid cross-entropy loss function. The height and width of all filters was set to 3 and the rectified linear unit [*Nair and Hinton*, 2010] was used as activation function. Furthermore, the standard deviation of the 3D Gaussian distribution was selected as 200 m. This is a hyperparameter that needs to be selected before training and it represents a trade-off between the resolution and training convergence. For more details about QNetSynth we refer to *Vinard et al.* [2022].

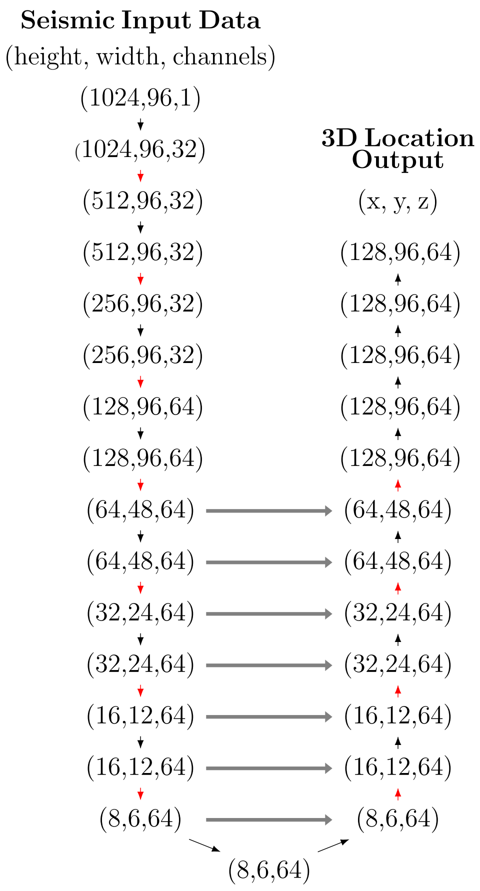


Figure 4.1: *DNN architecture with seismic data as input and 3D location output. Encoder (left) consisting of convolutional layers with red arrows denoting strided convolutions used for down-sampling. Decoder (right) with transposed convolutional layers with red arrows indicating strided transposed convolutions used for up-sampling. Gray horizontal arrows denote skip connections.*

■ 4.1.1 TL for post-monitoring and continuous acquisition mode

We investigate two possible applications, one that is suitable for post-monitoring processing and the other represents a continuous acquisition mode. In both scenarios alternative detection and localization methods are used to build a field training set that can be used to update the weights of QNetSynth. *Alexandrov et al.* [2020] detected and localized the events used in this study. As detection and localization algorithm the diffraction stacking (DS) algorithm [*Anikiev et al.*, 2014] was used and some of the DS-localized events were further improved in a post-processing step by a relative location (RL) method [*Grechka et al.*, 2015] that requires a set of master events. We refer to the latter method as DSRL. For the post-monitoring scenario we build a training set with a subset of the DS-detected events as input (waveforms) and we use the DS-locations to create the corresponding labels. QNetSynth is next updated using that training set and this updated network, QNet, is next applied to the remaining detected events. The QNet predicted locations are then compared to the DSRL-locations. In the continuous acquisition mode scenario the HF operations are still ongoing and events are detected and localized by DS. After the first day of operations a training set can be built from the DS-detected and DS-localized events, which is then used for TL to update QNetSynth. The updated DNN, QNet, can then be applied on the next day to retrieve locations of the DS-detected events. After each day the previous QNet can be updated with new DS-detected and DS-localized events and applied on the next day together with DS.

In order for the QNet to return good source locations on new data, the training set needs to be similar to the new data. This is a major challenge as source locations vary and change over time, especially in HF operations. Since the field data used for training may not cover all possible locations, the DNN will be biased towards the locations in the field data set used for training and fail to generalize to other locations. This would limit the applicability of the method. To overcome this issue we enrich the training set with synthetic data that were used to train QNetSynth and which cover the entire region of interest.

■ 4.1.2 TL Workflow

The general TL workflow is summarized by the flowchart in Fig. 4.2. We start the learning process with QNetSynth, which will be equal to QNetPrev (where Prev stands for previous) in the total TL flow. The training set is a combination of labeled field and labeled synthetic data. For the case of the labeled field data, the source locations were computed by another method (e.g. diffraction stacking). The synthetic database contains all events that were used to train QNetSynth. Instead of using all of the synthetic data we randomly pick a subset of n synthetic events and apply data augmentations (random bulk time shifts, random muting of traces, adding field noise) to the synthetics (see also *Vinard et al.* [2022]). The combination of the augmented synthetic data set with the field data forms the training set that is used to update the weights of QNetPrev in the TL process. Note that at each epoch (training set passed forward and backwards through the network to update the weights) a new subset of n random synthetics is selected from the synthetic database. The training set contains more labeled synthetic than field data, however the synthetic data used at each epoch is different due to random sampling of events from the larger synthetic database and random data augmentations, whereas the field data are always the same (we tested different values of n to determine its optimal value). If all of the synthetic data were used for TL at each epoch the field data would be underrepresented in the training set. This would lead to a highly imbalanced training set [*Chawla et al.*, 2004] and the weights in the updated QNet would be biased towards the synthetic data with few changes in its weights in favor of field data. After TL, QNet can be used to reconstruct 3D Gaussian distributions on new field data, where the waveforms are taken as input and the output is a 3D distribution as shown in Fig. 4.3. The TL process can be repeated whenever an updating condition is met. In that case the field data set used in training is enlarged with new events and QNet is set as QNetPrev to repeat the TL process with the newest set of weights. Again, here we take the source locations determined by another method (e.g. DS) to create the labels for the new set of events.

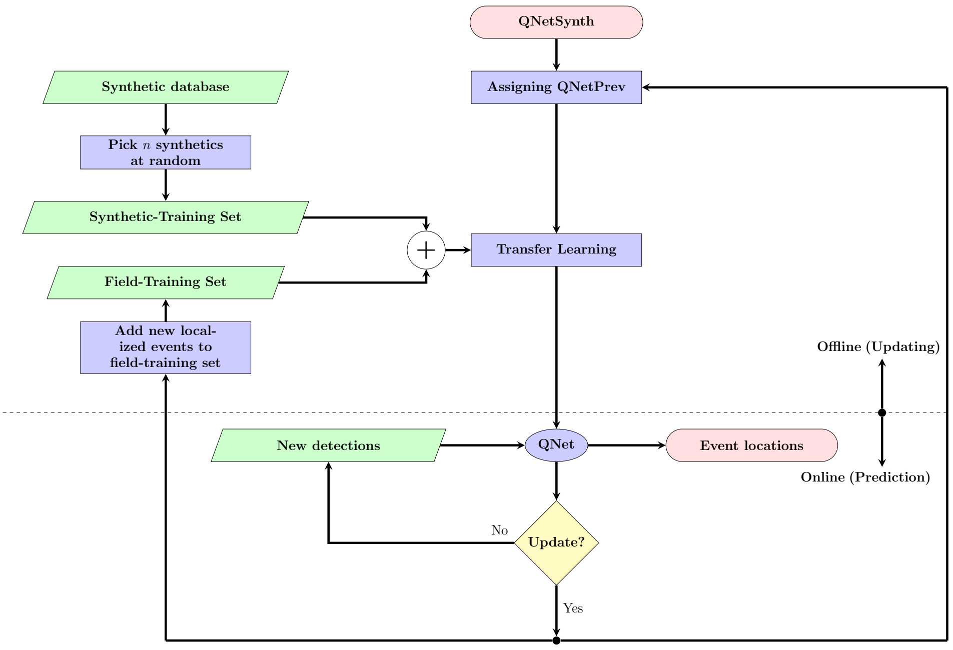
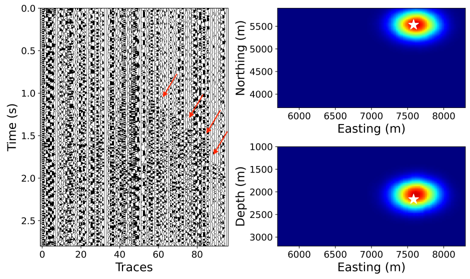
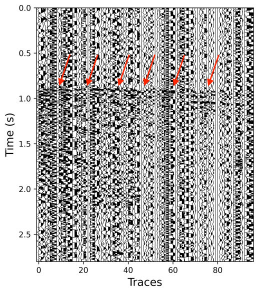


Figure 4.2: *Workflow describing TL to update weights of QNetPrev, applying QNet on new field data and possible further updating of QNet using more field data.*

Updating QNetPrev only with the field data set could lead to overfitting, meaning that QNet can produce very good results on the training data but fail to accurately locate events in areas where there were no samples in the training data. Combining TL with both the field and the synthetic data helps to reduce overfitting.



(a)



(b)

Figure 4.3: *(a) Example of waveform input to QNet on the left and its output (cross sections taken at maximum voxel in map view and view from south) on the right. White star denotes location computed by DS. Clipping applied for visualization purposes only. (b) Normal-moveout corrected version of the input waveform, created using the 1D velocity model used for DS, to better visualize signal.*

The data augmentations applied to the synthetic data are used to increase the size of the data set and also help to reduce overfitting. Random bulk time shifts and random muting of traces are augmentations that are easy to implement and also help the learning process by creating variability in the data. The addition of field noise on top of the synthetics was shown to be crucial for QNetSynth to localize field data events [*Vinard et al.*, 2022].

■ 4.1.3 TL training and evaluation

During TL we allow all the weights of QNetPrev to change. Freezing parts of the weights during TL, i.e. preventing those weights to be updated during TL, did not result in noticeable changes. Thus, we decided to allow all of the weights to change. We use the Tensorflow software [*Abadi et al.*, 2015] for training using the Adam optimizer [*Kingma and Ba*, 2015]. We set the learning rate (step size) to 0.001 and use a batch size of 20 (number of training examples that are passed forward and backward through the network to update the weights). The same learning rate and batch size were used to train QNetSynth [*Vinard et al.*, 2022]. Note that the input data (waveforms) are always (not only during training) normalized by their maximum amplitude value before being passed to the QNet.

During TL we set a fixed number of epochs over which to update the weights of QNetPrev. We compute a metric between the output generated by the DNN and the expected 3D Gaussian distribution on the validation set at the end of every epoch and after training we save the weights that maximized that metric over all epochs. As metric we use the Dice similarity coefficient (C_{Dice}) [*Dice*, 1945], which is defined as,

$$C_{Dice}(g,\hat{g}) = 2\frac{g \cap \hat{g}}{g + \hat{g}}, \qquad (4.1.2)$$

where g is the label, i.e. the 3D-Gaussian distribution defined with its peak at the location given by the DS, and $\hat{g}$ is the 3D output distribution produced by the QNet. As in *Vinard et al.* [2022] we clip the values in both g and $\hat{g}$ above 0.1 to 1 and the rest to 0 before computing the C_{Dice}. Thus, if there is a perfect overlap the C_{Dice}-value equals one and if there is no overlap it is zero. As loss function we use the sigmoid cross-entropy loss that was also used to train QNetSynth.

4.2 Data

The field data used in this study were acquired in Texas, USA, in 2010, during hydraulic-fracturing operations in the Barnett Shale Formation in the Forth Worth basin. The monitoring system used 543 vertical component geophones buried in shallow boreholes where each borehole contained three geophones placed at 30, 45 and 60 meters below the surface. The system covered an area of approximately 144 square kilometers. *Alexandrov et al.* [2020] generated a 1D layered P-velocity model from the site from sonic logs and computed hypocenter locations of the events using a migration-type diffraction-stacking (DS) technique [*Anikiev et al.*, 2014] and further refined the location of some events using a relative location (RL) method [*Grechka et al.*, 2015] using a set of 27 master events. The diffraction stacking and relative location, DSRL, method improved the depth estimates of the events, relocating them closer to the injection wells located between 2000 and 2200 meters depth. However, the relative locations can only be computed after the whole monitoring period since it requires a set of master events, which are usually only available for postprocessing. This is why the DS-locations are used to create the labels during training. However, after training the DNN's performance is compared to the DSRL-locations.

The S/N of the events in our dataset are very low with the majority of events having S/N below 1 dB, as summarized in Table 4.1. As Table 4.1 reveals 622 events have a S/N ratio below 0.77 dB, implying that on most traces the signal is below the noise level. The signal for such events is enhanced by diffraction stacking allowing those events to be detected and located.

Table 4.1: *S/N statistics of the 1245 field data events in dB, as taken from the S/N computed by [Alexandrov et al., 2020].*

Mean	Std.	Min.	25%	50%	75%	Max.
0.87	0.55	0.40	0.64	0.77	0.96	9.31

4.2.1 Synthetic data

QNetSynth was trained with synthetic data modeled with the reflectivity method [*Kennett and Kerry*, 1979] with the software ERZSOL3 [*Kennett*, 2005] using the layered P-velocity model generated by [*Alexandrov et al.*, 2020] and with the same geophone locations defined in [*Vinard et al.*, 2022]. For QNetSynth only 96 geophone locations were used mainly to reduce computational and memory costs. QNetSynth was trained with 51200 synthetic events that cover the entire event region. This region as well as the receiver locations and well locations are shown in Fig. 4.4. Within the region of interest random double-couple sources with center-frequencies ranging between 20-24 Hz were modeled. The synthetics were augmented with field noise, Gaussian noise varying in amplitude per trace, random bulk time shifts and random muting of traces during training. The size of the input data is (1024 x 96 x 1), and the 3D region where events can occur is discretized to a shape of (128 x 96 x 64) with grid size of (23 m, 27 m, 34 m) in Easting, Northing and depth, respectively. The label of each event is defined by its 3D Gaussian distribution with the peak equal to 1 at the source location and with a fixed standard deviation of 200 m in all directions. This fixed standard deviation was chosen for QNetSynth and found to be optimal for good convergence and resolution [*Vinard et al.*, 2022]. For more details about the synthetic data we refer to *Vinard et al.* [2022].

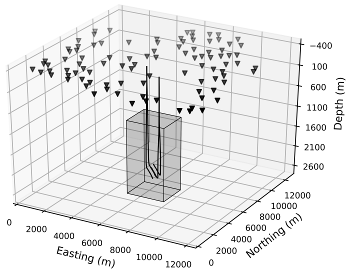


Figure 4.4: *Receiver locations (black triangles) and region where events can occur (shaded cuboid) extending from 5700 to 8300 m in Easting, 3700 to 5900 m in Northing and 1200 to 3000 m in depth. For the simulations this space is increased by 200 m in all directions. Black lines represent orientation of wells.*

■ 4.2.2 Field data preprocessing

As for the synthetic data, the label for the field data for training is created in the same way, i.e. as a 3D Gaussian distribution with a standard deviation of 200 m. For the field data we use the locations retrieved by DS to create the Gaussian distribution. We apply a band-pass filter of 5-50 Hz to the detected field data (during both training and application on new data). No denoising steps are performed to the data. Finally, note that all of the data used to train the DNN and to make predictions were previously detected and confirmed as true detections by *Alexandrov et al.* [2020].

■ 4.2.3 Field data for post-monitoring application

In order to apply transfer learning we need labeled field data. Thus a preprocessing step that detects and localizes a number of events is needed. In our case the preprocessing step was carried out by *Alexandrov et al.* [2020] and 1245 events were detected and located from 9 days of monitoring. We divide these events into a field-training, validation and test sets. The labels are created using the DS-locations. The field-training and validation sets are used during TL to update the weights of QNetSynth and to determine the weights that maximized the C_{Dice} (eq. 5.3) on the validation set over all epochs. Finally, the test set is used to apply the updated QNet to data not used during TL.

We randomly split the 1245 events using 60% for training, 20% for validation and 20% for testing. Thus 747 events serve as a field-training set and 249 events each as validation and test sets. This partitioning of the data is used for the post-monitoring application. The DSRL-epicenter locations of the 1245 events are shown in Fig. 4.5 together with the well locations.

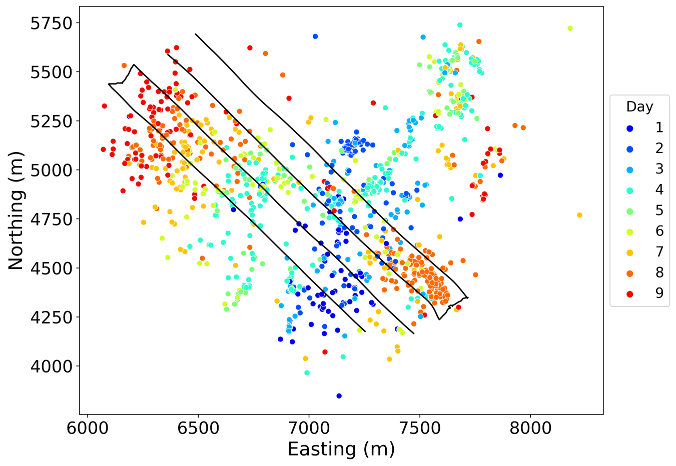


Figure 4.5: *Epicenter locations of 1245 DS-detected and DSRL-localized events from first to ninth day of monitoring. Each colored dot represents an event recorded on a particular day. Well positions are shown by black lines.*

■ 4.2.4 Field data for continuous acquisition modes

The methodology for continuous acquisition modes is based on dividing the time into intervals (in our case study into days) and using the labeled data from past intervals in TL for the newest time intervals. In this case study, as new events are

DS-detected and DS-localized we use those to update the weights of the current model and apply it to events that are DS-detected the following day. Thus, after the first day of monitoring we update QNetSynth with the DS-detected and DS-localized data from the first day and apply the updated QNet to localize the DS-detected events from the second day of monitoring. Next, we update the QNet after the second day of monitoring with field data events that were DS-detected and DS-localized during the first two days of monitoring. This is repeated up until the last day of monitoring. Thus, the TL process summarized in Fig. 4.2 is looped once per day. The events detected each day are randomly split into a field-training and validation set with a 80/20% ratio. The field-training data set is used to update the weights of the network in combination with the synthetic data for a fixed number of epochs. At the end of each epoch we compute the C_{Dice} over the entire validation set and after training we select the weights from the epoch that maximized the C_{Dice}. It is important to regularly update the DNN due to changing event locations that can affect the performance of the DNN. The changes in DSRL-epicenter locations from the first to the ninth day of monitoring (indicated by colors) are shown in Fig. 4.5. It can be observed that the event locations change over time.

4.3 Results

We present the results of both the post-monitoring and the continuous acquisition mode source-localization applications. Starting with QNetSynth we apply TL to update its weights using a combination of field and synthetic data.

■ 4.3.1 Post-monitoring application

During TL we use the 747 field data events and synthetic data as training data. As mentioned above the labels are created using the locations recovered by DS. However, we compare the QNet predicted locations to the DSRL-locations as those are the more accurate locations. We use 100 epochs for training and choose the weights from the epoch that maximizes the C_{Dice} (eq. 5.3) on the validation set and name the updated DNN, QNet. Increasing the number of epochs did not bring any significant improvements. To create the training set a limited number of random synthetics were selected from the synthetic database at each epoch. The synthetic database contains 51200 events, from which we randomly select a subset at each epoch. Thus, at every epoch a new set of synthetic samples are picked and added to the field-data events used for training. We experimented with different numbers and got best results by randomly sampling 2000 events from the synthetic database at each epoch.

After TL QNet is ready to be applied to the test set. The test set contains 249 events with moment magnitudes between -0.59 and 1.52. In Fig. 4.6 we show the locations retrieved by QNet from the peak of the reconstructed distribution for all 249 events in the test set compared to the DSRL-locations. Note that a grid pattern emerges in the DNN predicted locations, which is due to the discrete 3D output space. In general the hypocenter locations returned by QNet match well with the DSRL-localized events. The mean distances in hypocenter, epicenter and depth between the locations provided by QNet and DSRL are 227 m, 148 m and 141 m, respectively and with a median hypocenter distance of 182 m. The depth locations returned by QNet are concentrated in depth between 2000 and 2200 meters for most events. This is also the depth of the fractured interval.

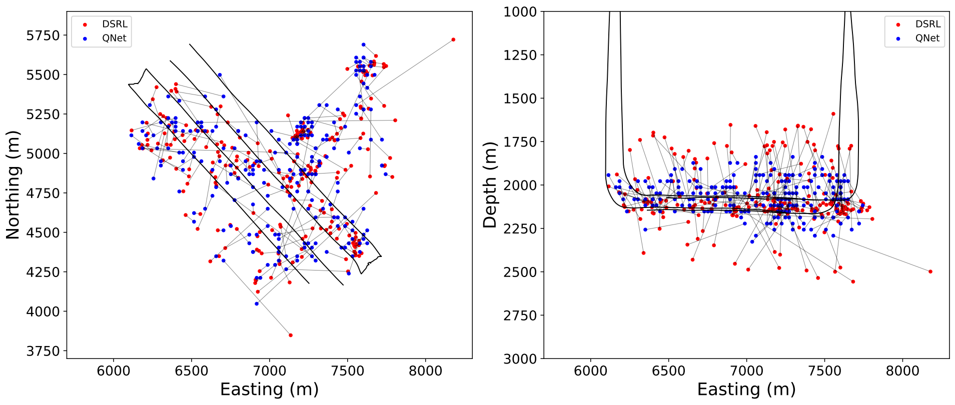


Figure 4.6: *Locations predicted by QNet on test set for post-monitoring processing application plotted together with DSRL-locations. Lines connect locations of same event predicted by QNet and DSRL. Left-side plot is map view and right-side plot is view from south.*

The histogram (Fig. 4.7) of the location errors, computed as the distances between QNet-located events (from the peak of the distribution) and the DSRL-locations shows that a majority of the events are located within 300 m from each other with a sharp decrease in events with distances greater than 300 m away from the DSRL-locations in the validation and test sets. We take a closer look at those latter events. We compare their magnitudes and S/N's with the events located less than 300 m from the DSRL-locations in the test set as well as to the magnitudes and S/N's present in the field data used for training. We only plot the magnitudes between -0.6 and 1.0 and S/N between 0.4 and 1.9 in order to better observe the events that were located at greater distances from the DSRL-locations, see Fig. 4.8. We observe that a majority of the events that are located further than 300 m from each other have low magnitudes and S/N. This is to be expected since there are less events within that moment magnitude and S/N range in the field-training set and thus fewer of those examples that the DNN can learn from.

■ 4.3.2 Continuous acquisition mode

For the continuous acquisition mode we update the DNN on a daily basis, starting with the DNN that was trained on synthetic data, QNetSynth. For labeling of the detected events we use the DS-locations.

To investigate the added value of the synthetic data, we apply TL excluding and

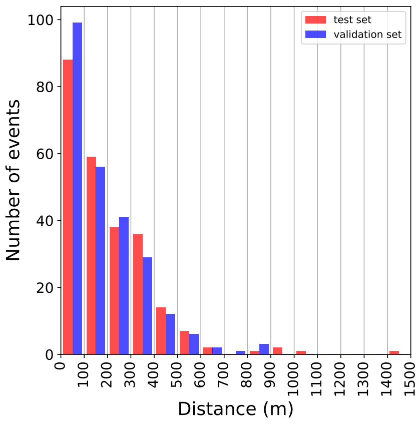


Figure 4.7: *Histogram showing number of events located in different distance bins of 100 meters width as a function of the distances to the DSRL-locations in test and validation set.*

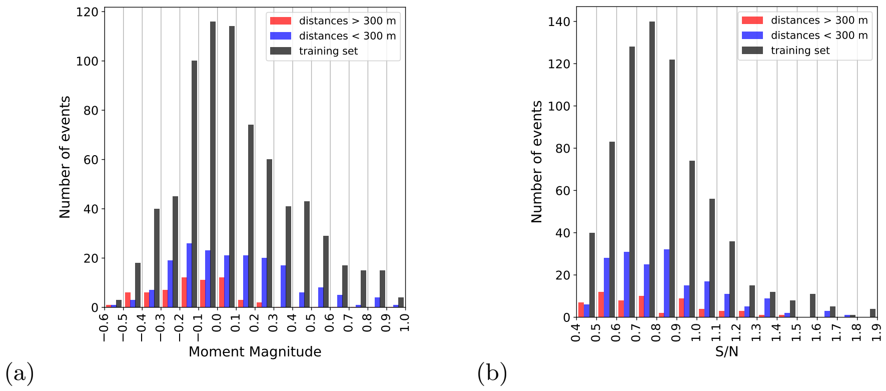


Figure 4.8: *Histograms of moment magnitudes (left) and S/N (right) of events predicted at distances greater than 300 m (red) and less than 300 m (blue) from DSRL-locations in the test set and distributions in training set (black).*

including synthetic data. Starting with QNetSynth and the events detected and located after the first day of monitoring, we follow the scheme described in Fig. 4.2: QNetSynth becomes QNetPrev and the field-data events detected and localized the preceding day by DS are used to update the weights of QNetPrev. For the situation where synthetic data are also used during TL we randomly select a new set of 2000 synthetic events at each epoch, as we did for the post-monitoring application. We randomly create splits of 80/20% of the field data events to serve as field-training and validation sets, respectively. After training for 100 epochs, we again keep those weights that maximized the C_{Dice} on the validation set. This updated model, QNet, is next applied to data detected on the next day of monitoring. This process is repeated until the last day of monitoring. We refer to the QNets obtained with this iterative TL workflow without synthetics as QNet1 and those updated with synthetics as QNet2. The C_{Dice}- and loss-curves over the training and validation set using data after the first day of operations are shown in Fig. 4.9. The curves look similar for the remaining TL iterations. The vertical dashed line shows the epoch at which the C_{Dice}-value over the validation set reached its maximum value. The training- and validation-loss curves are close to each other, indicating that the model is not heavily overfitting and the validation loss in Fig. 4.9 is steadier after 70 epochs.

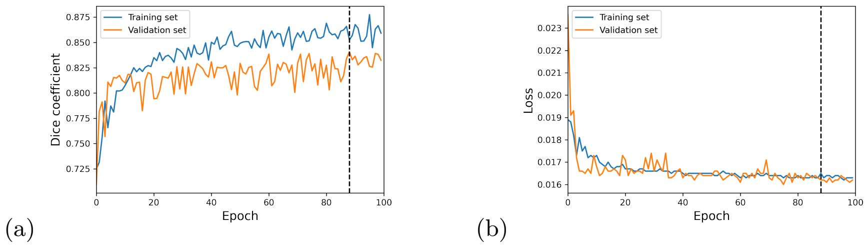


Figure 4.9: *C_{Dice}-curves (left) and loss-curves (right) of training and validation set during first TL iteration in continuous acquisition mode.*

After each TL iteration we apply the updated QNets, QNet1 and QNet2, to the same field data detected the following day and compare the results to the DSRL-locations of the events. Fig. 4.10 shows the epicenter locations from the second to the ninth day of monitoring separately for QNet1 and QNet2. The red dots are DSRL-localizations and the blue dots represent locations retrieved by the QNets (QNet1 in the first and the third column and QNet2 in the second and the fourth column). The lines connect the DS-localizations to the locations predicted by the QNets.

With the exception of a few events the epicenter locations recovered by QNet2 on the second day of monitoring better compare to the DSRL-locations compared to QNet1. The DNN updated without synthetic data during TL, QNet1, mislocates the small cluster of events located in the upper part of the plot (black circle in Fig. 4.10). A similar observation can be made for the epicenters from the third day of monitoring where QNet1 does worse at localizing the small clusters on the upper and lower parts of the plot whereas QNet2 does much better with exception of a few outliers. These problems can be explained by the lack of training samples in those areas. This can very clearly be observed on the fourth day of monitoring when comparing the locations predicted by QNet1 to those of QNet2. As the event locations up to the day used to update QNetPrev are always slightly different to the event locations from the following day, QNet1 seems to always be lagging behind

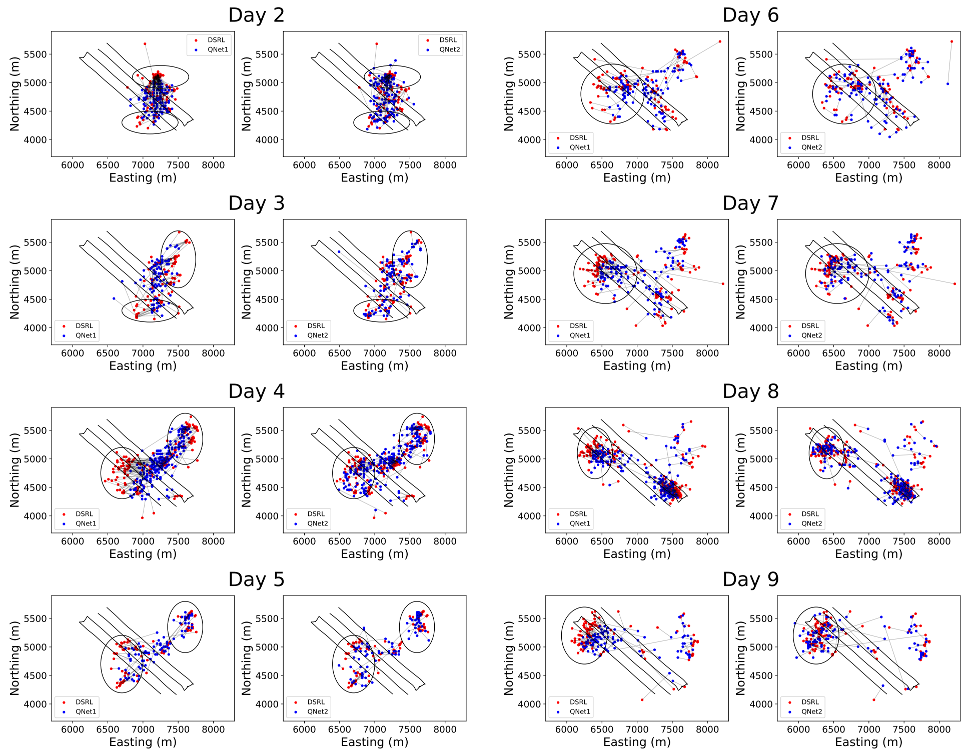


Figure 4.10: *Epicenters returned by QNet1 updated without synthetics (columns 1 and 3) and QNet2 updated with synthetics (columns 2 and 4) from second to ninth day of monitoring. DSRL and respective DNN locations (QNet1 and QNet2) connected by black lines.*

a little, as also observed throughout days 5 to 9 of monitoring. QNet2, however, can overcome this issue due to the use of the synthetic data, which well samples the locations of interest.

The mean distance between all locations predicted by QNet1 and the DSRL-locations is 282 m. The mean epicenter distance is 226 meters. For QNet2 the mean hypocenter distance is 249 m and the mean epicenter distance is 167 m. These differences are large, however, note that DS-locations and not DSRL-locations were used for training and furthermore, we cannot be sure that the DSRL-locations are true locations. Thus, updating QNetPrev with the synthetic data that covers the entire event region is more consistent with the DSRL-localized events. This is especially important with continuous processing where new events occur in regions where past events did not occur and were therefore not part of the training set used for TL. Fig. 4.5 shows how the events migrate on a daily basis. The depth differences between the DSRL-locations and the predictions by QNet1 and QNet2 are similar with 146 m and 151 m, respectively.

4.4 Discussion

In the introduction we mentioned that QNetSynth (trained purely with synthetics) failed to localize many of the lower magnitude field-data events. In order to show how TL helps to improve the localization of field data events, we compare the locations of QNetSynth with the daily updated versions of QNet2, applied to the events recorded from the second to the ninth day of monitoring. We plot the moment magnitude of the events versus the distance between the DSRL locations and the locations returned by both QNet2 and QNetSynth in Fig 4.11a. We can see that QNet2 is able to localize many of the lower magnitude events more accurately than QNetSynth.

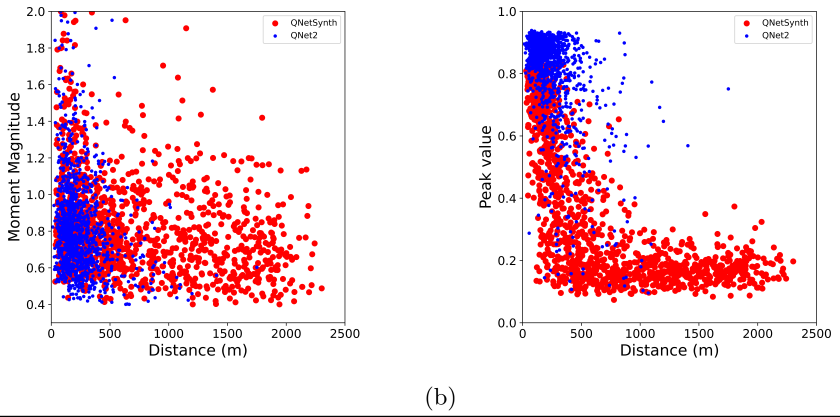


Figure 4.11: *Comparing QNetSynth with QNet2. (a) Moment magnitude vs distance and (b) peak value vs euclidean distance between the QNet-locations and DSRL-locations. QNet-Synth displayed by large red dots and QNet2 by small blue dots.*

We consider the maximum value of the distribution to be at the source location and refer to it as the peak value. We can see from Fig. 4.11b, where the peak value of the output is plotted with respect to distance, that the peak values are higher for QNet2. Thus by updating QNetSynth with field data (and synthetics) the number of confidently localized events increases. Based on Fig. 4.11b we might consider setting a threshold on the peak value to only accept events that are above it. In practice the threshold should be based on the validation set. For now we set a threshold to 0.5 for both QNetSynth and QNet2 and compare the moment magnitude distribution of the events that pass the threshold (Fig. 4.12). We can see that most of the higher magnitude events for QNetSynth passed the threshold but that many of the lower magnitude events did not. Between moment magnitudes 0.6 to 1.6 roughly the same number of events pass the threshold for QNetSynth and QNet2. For magnitudes below 0.6 increasingly more events pass the threshold in the

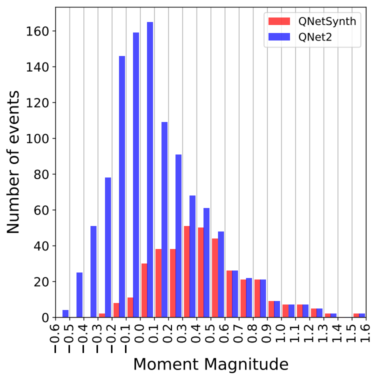


Figure 4.12: *Magnitude distributions of events passing threshold 0.5 for QNetSynth (red) and QNet2 (blue).*

case of QNet2 compared to QNetSynth and no events with **M** below -0.3 pass the threshold for QNetSynth whereas for QNet2 events down to **M** between -0.6 to -0.5 pass the threshold. The mean distance over all events recorded after the first day of monitoring is 688 m for QNetSynth and 249 m for QNet2. The localization improved similarly comparing QNetSynth to QNet: Over the test set of 249 events the mean distance between QNetSynth locations and DSRL-locations is 747 m, whereas for QNet the mean distance is 227 m.

In this study we focus our attention on the locations of detected microseismic events. For a practical situation a detection algorithm needs to be employed to first detect an event. In this study the diffraction stacking algorithm was used for detection [*Anikiev et al.*, 2014]. Alternatively, several machine learning based seismic event detection algorithms have been successfully applied in recent years [*Perol et al.*, 2018; *Wu et al.*, 2018b; *Meier et al.*, 2019; *Wu et al.*, 2018b; *Dokht et al.*, 2019; *Mousavi et al.*, 2019].

During the supervised learning phase our DNN was only trained with data containing events and it learned to extract features from that input and map it into a 3D Gaussian distribution. Thus, the QNet was not fully trained to differentiate between noise and seismic events. However, if the input to the QNet contains noise only, we do not expect it to return a 3D Gaussian distribution with a high peak-value. Therefore, the QNet might serve as an event detector. To investigate this possibility we take a time window of approximately 7 s around an event recorded on the 6th day of monitoring. Next, we pass QNet2 (trained with iterative TL up to day 5) chunks of 2.8 s each shifted by 0.6 s from start to finish. The seismic data that is used as input to QNet2 as well as its output are shown in Fig. 4.13. The predicted output is sliced horizontally and vertically through the maximum output voxel. The peak of the output distribution corresponding to noise is significantly below 1 before the signal enters the time-window and the output can not be characterized as Gaussian. As soon as the signal appears on the first few receivers, the distribution's peak value is significantly higher and resembles a Gaussian distribution. However, the peak of the distribution does not yet match with the DS-location. In the two consecutive time windows the peak of the distribution is on top of the DS-location and the peak value is at its highest. Finally, at the later time steps, as the first signals start passing the receiver array, the distribution starts to change and eventually dissipates as no signals are present in the input. We believe this methodology can be extended to provide detection of seismic events.

The synthetic data used to train QNetSynth and also used in the TL scheme was generated using the same velocity model as used to localize the events by diffraction stacking. The velocity model may not always be well known. If the velocity is complex then to create the synthetics we may need to use a more computationally intensive method to compute seismograms. Therefore, to analyze how accurate the velocity models need to be in order to train a DNN that is able to provide good locations for field data is a recommendation for further research. As is known from other methods, the accuracy of the locations depends on the velocity model and we expect this to be the case for this method as well.

In this study we benefited from a well known velocity model. However, in general the velocity model, especially if used for simulating full waveform synthetic seismograms, may not be well known. Further investigation on accuracy of the velocity model may help us to understand limitations of the proposed methodology, but this is beyond the scope of this study as we need to define the quality of the velocity model to judge better.

When creating our training, validation and test sets from the field data we randomly created the splits. Thus all three sets roughly cover the same moment magnitudes. It would be interesting to investigate the possibility of applying TL using high magnitude events in a first run and apply the updated DNN to low magnitude events to test if it can extrapolate its feature extraction and classification capacities to the those events.

In order to get an idea about the computational effort needed for our approaches: to generate 51200 synthetics and running 100 simulations in parallel takes roughly 7 days on 2.3 GHz Intel Xeon CPUs. The training time depends on the size of the training set. A single epoch took approximately 60 seconds on a Tesla P100-PCIE-16GB GPU. Thus training for 100 epochs takes about 1.7 hours. Finally, applying the trained QNet on a single event to generate the output takes 0.28 seconds on a 3.1 GHz Dual-Core Intel Core i5 CPU. Hereby we show that it is feasible to apply this method on a daily basis.

4.5 Conclusions

In this work we introduced a TL scheme to update a DNN previously trained on synthetic data. The TL scheme can be either used a single time in a post-monitoring situation or iteratively for continuous monitoring. In the TL scheme the QNet is updated using a combination of labeled field and labeled synthetic data. By updating the QNet in this fashion the number of confidently localized field-data events at low magnitudes drastically increased. Furthermore, we showed the importance of

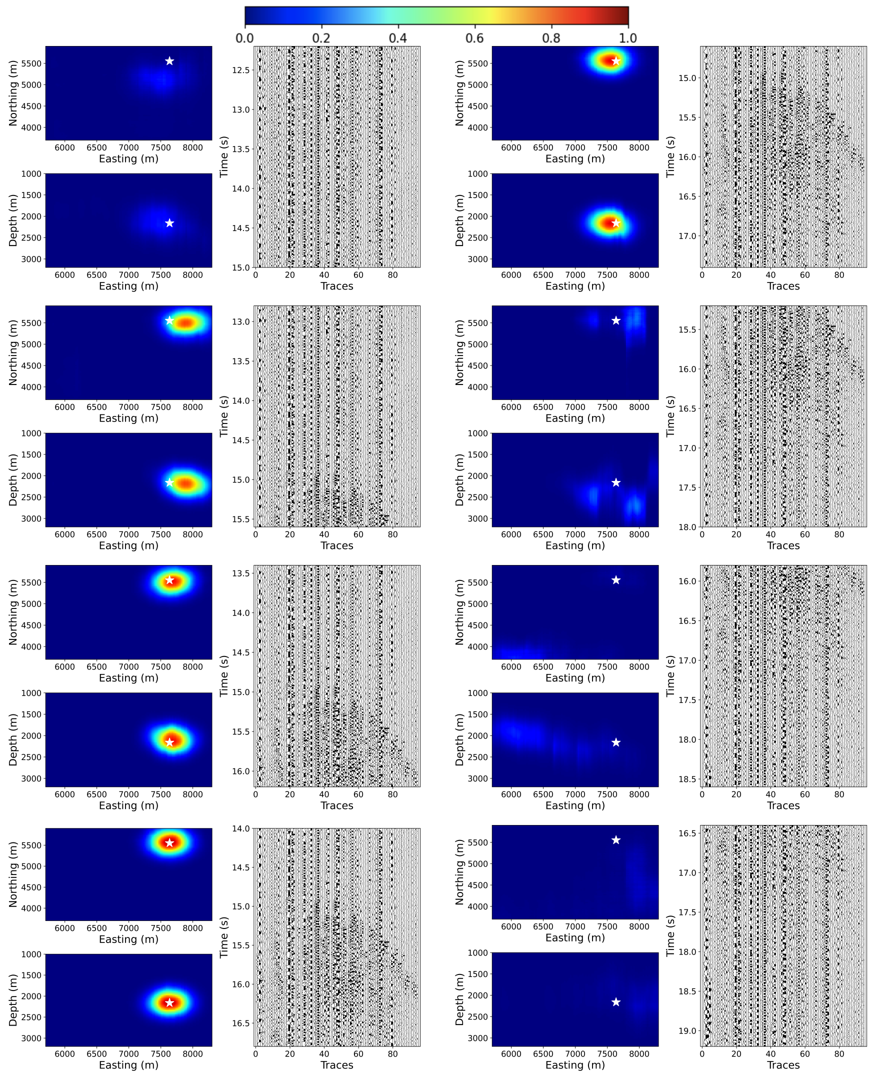


***Figure 4.13:** QNet2 trained on first 4 days of field data applied to event recorded on 6th day. QNet2 receives seismic data as input (records) and returns 3D output (plan and section slice through maximum value). White star denotes DS-location. Time increases from left column downwards and continues from upper left column. Clipping applied to better visualize events in records (not applied during training and prediction).*

keeping the synthetic data during TL in order to provide accurate source locations in areas not yet covered by the field data used during training. Additionally, we provide a framework to regularly apply TL in a continuous data processing mode, which increases the localization performance of the QNet over time.

5
Fine tuning a deep neural network to localize low magnitude earthquakes

"Learn to deal with the valleys and the hills will take care of themselves."

Count Basie, 1926

Abstract A main challenge in microseismic monitoring is that the seismic signals recorded at the Earth's surface are weak and thus localization of those microseismic earthquakes becomes challenging. Diffraction stacking is a traditional method used to localize weak earthquakes, which involves stacking the waveforms along precomputed travel-time curves from different locations, where the maximum is used to determine the source location. In this work we aim to recover the source location of weak microseismic earthquakes using a deep neural network (DNN) that resembles the U-Net but uses fewer skip connections. However, the size of the field data is too small to train the DNN from scratch. Thus, we propose to pretrain a DNN using synthetic data that resembles the field data and that learns to map the source location in terms of a 3D Gaussian distribution directly from the seismic signals. This pretrained DNN is capable of localizing the higher magnitude earthquakes in the field data, but fails for the weaker earthquakes. To be able to localize the weaker magnitude earthquakes we therefore, fine tune the pretrained DNN using the higher magnitude field-data earthquakes. We observe that the updated model is able to extrapolate the information learned during the fine tuning step from higher magnitude earthquake data to lower magnitude earthquake data.

Published as: N. A. Vinard, G. G. Drijkoningen and D. J. Verschuur (2021), Fine tuning a deep neural network to localize low magnitude earthquakes, IEEE ICECET Proceedings.
Note that minor changes have been introduced to make the text consistent with the other chapters.

5.1 Introduction

Industrial activities in the subsurface such as waste-water injection, hydrocarbon extraction, geothermal stimulation, CO_2-sequestration and hydraulic fracturing generate stress changes in the subsurface, which lead to primarily small earthquakes, known as microseismic events [*Foulger et al.*, 2018]. In order to better understand where these changes in the subsurface occur, the location of the microseismic events can be determined from data collected by seismic sensors placed closed to the Earth's surface surrounding the area of the industrial activities. Since most of those events have low magnitudes their signals are weak and thus they are difficult to detect and localize. To detect and localize microseismic events from seismic signals, array-processing methods, which most often take the wave propagation effects into account, are used. One such method is diffraction stacking (DS), which stacks the waveforms along precomputed travel-time curves [*Anikiev et al.*, 2014]. This involves a grid search over all possible source locations, where the most likely source location can be based on different conditions, such as the maximum over all stacks. Another difficulty is caused by the complex radiation pattern of microseismic events, resulting for instance in different polarities at different sensor locations. Thus, for optimal stacking, the polarities can be corrected by also applying a grid search over different source mechanisms.

Deep learning methods and especially convolutional neural networks have been widely applied to seismic data for many different applications, e.g. phase detection [*Zhu and Beroza*, 2019], earthquake signal detection [*Mousavi et al.*, 2019], earthquake magnitude estimation [*Lomax et al.*, 2019] and source localization [*Kriegerowski et al.*, 2019; *Mousavi and Beroza*, 2020a; *Van den Ende and Ampuero*, 2020; *Zhang et al.*, 2020b]. Most of these source-localization methods trained their models on field data by splitting the data set into a training, validation and test set. For small field data sets it may not be possible to train a deep neural network (DNN) from scratch as usually large data sets are needed to train deep learning models. In such situations it is possible to use weights from another DNN that was trained for a similar task to avoid starting training from scratch and to fine tune those weights on a smaller data set. One such example in a seismic application used a model trained for seismic phase picking of earthquakes [*Zhu and Beroza*, 2019] to fine tune it for picking phases of microseismic events [*Chai et al.*, 2020]. Others used large synthetic data sets to pretrain their models and then fine tuned the models on smaller field data sets for interpolating seismic traces [*Wang et al.*, 2020] and for seismic fault detection [*Cunha et al.*, 2020].

In this work we aim to first train a modified U-Net [*Ronneberger et al.*, 2015] (a convolutional neural network developed for biomedical image segmentation) on a large synthetic data set and next fine tune the model on higher magnitude field data events in order to apply it to lower magnitude events. The synthetic data set is generated using a known layered velocity model from the monitoring area, the expected region where events are likely to occur and the known sensor locations given in the field. The need for a synthetic data set is due to the small size of the field data set, which makes it difficult to start training from scratch. However, this is not the only purpose of the synthetic data set. The synthetic data set is also useful because it covers almost all possible source locations and thus, the model trained on the synthetic data set, learns to associate input data from many source locations. Since the synthetic data do not perfectly resemble field data, we fine tune the synthetically trained network with high magnitude field data events. The fine-tuned model is able to accurately localize lower magnitude field data events that the synthetically trained model was unable to localize. The case study data uses microseismic events recorded during hydraulic fracturing operations.

In contrast to previous work in the field of earthquake localization, we first train our network on synthetic data instead of field data. Secondly, we fine-tune the synthetically trained model only with the higher magnitude field data events instead of sampling from the entire field data population, to investigate the capacity of the model to expand its feature extraction capacities beyond the data encountered in the training set.

The main goals of this work are to (1) determine the location of microseismic earthquakes and to show the value of synthetic data for situations where (2) the amount of field data is limited to train a DNN from scratch, (3) the observed field data do not cover the whole area where future events can occur and (4) synthetic data are relatively quickly generated and (5) to investigate the capacity of fine tuning the model on high magnitude events to localize low magnitude events.

5.2 Data

5.2.1 Field data

Our case study is based on microseismic field data recorded in 2010 in Texas, USA, during hydraulic fracturing operations [*Alexandrov et al.*, 2020]. From this monitoring period, we have 1245 events in our data set from nine days of operations. The source locations of those events were computed using a diffraction stacking (DS) method [*Anikiev et al.*, 2014] by [*Alexandrov et al.*, 2020], which we also have at our disposal. While the data were acquired by over 500 vertical-component sensors, we only used a subset of 96 sensors in our case study, in order to reduce the computational footprint. The 96 sensors as well as the region where events are expected to occur are shown in Fig. 5.1. We bandpass filter the field data in the frequency range 5-50 Hz. During both training and prediction the input data are first normalized by the maximum amplitude value in a time window.

5.2.2 Synthetic data

We simulated synthetic data given the locations of the 96 sensors, a P-wave velocity model of the subsurface and the region where events are expected to occur (ranging from 3700 to 5900 m in Northing, 5700 to 8300 m in Easting and 1200 to 3200 m in depth). Within the expected region we randomly generated 51200 sources, where each source was assigned a random location and source mechanism, which defines the radiation pattern of the source. To model the synthetic data we used the open source software ERZSOL3 [*Kennett*, 2005], which is based on the reflectivity method

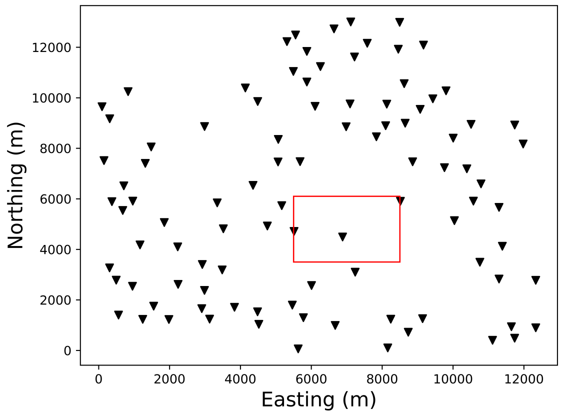


Figure 5.1: *Map view of sensor locations (triangles) and expected source region (red rectangle). Easting and Northing are relative locations.*

[*Kennett and Kerry*, 1979].

5.3 Methodology

The deep learning network that we use to recover the source locations given the seismic waveforms as input, resembles the U-Net architecture [*Ronneberger et al.*, 2015]. This architecture is suitable to map seismic input into a 3D Gaussian distribution by the use of the encoder-decoder architecture. The encoder is composed of a set of convolutional layers and the decoder is composed of a set of transposed convolutional layers. In the encoder the DNN extracts the features in the input that contain relevant signals, whereas the decoder maps the extracted features into a 3D Gaussian distribution with its peak defining the source location, similar to [*Zhang et al.*, 2020b]. The standard deviation of the Gaussian distribution is fixed, independent of the input data and it is selected before training based on the desired resolution and convergence speed of the training. The architecture with the shapes of each layer as well as an example of an input and output is shown in Fig. 5.2. Strided convolutions are used for downsampling in the encoder and, similarly, strided transposed convolutions are used for upsampling in the decoder. Instead of using skip connections through all hidden layers of the network, we only use them in a few layers, since we map the input to a different output space. We use the skip connections as they help speed up convergence [*Li et al.*, 2017]. The convolutional and transposed convolutional layers are followed by the ReLU activation function [*Nair and Hinton*, 2010] and batch normalization [*Ioffe and Szegedy*, 2015], except for the final transposed convolutional layer generating the output, where the sigmoid activation function is applied. The sigmoid cross-entropy is used as loss function,

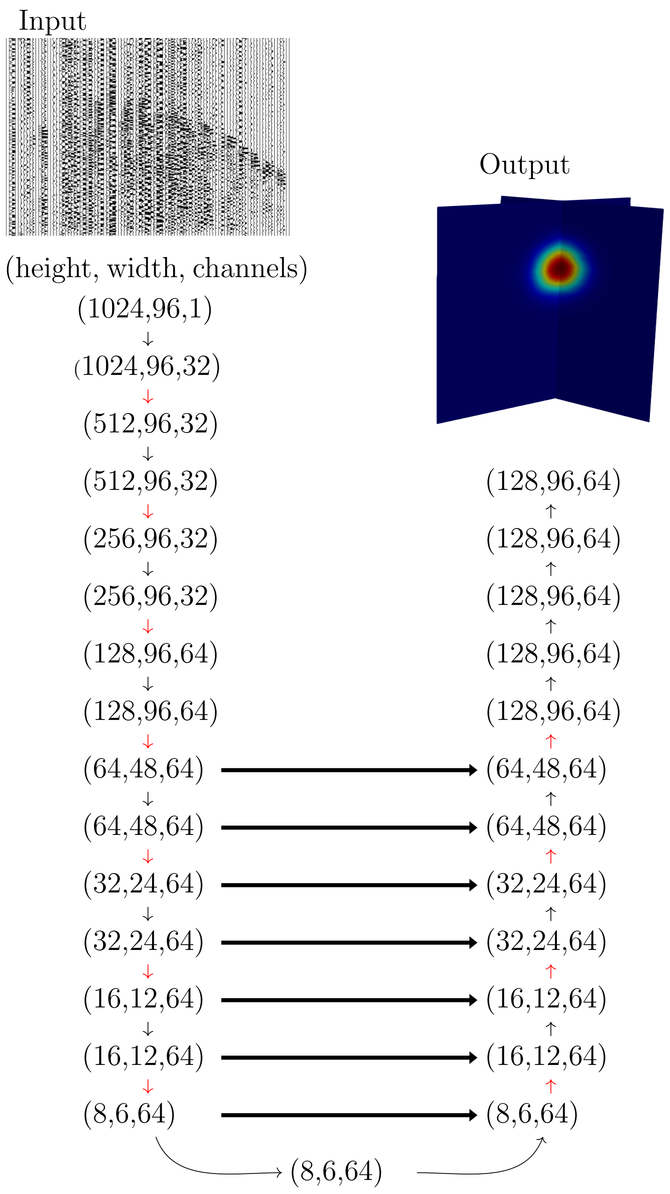


Figure 5.2: *DNN architecture with input (seismic field data), label (3D Gaussian distribution) and shapes. Skip connections denoted by thick horizontal arrows. Encoder with convolutional layers on left side, strided convolutions (red arrows) without strides (black arrows), and decoder with transposed convolutional layers on right side, strided transposed convolutions (red arrows) without strides (black arrows).*

$\mathfrak{L}$,

$$\mathfrak{L} = -\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left[g_j^{(i)}\log(\hat{g}_j^{(i)}) + (1-g_j^{(i)})\log(1-\hat{g}_j^{(i)})\right], \tag{5.3.1}$$

with g_j being the label of the j-th input and $\hat{g}$ being the output returned by the DNN, M stands for the number of voxels and N for the number of training samples the loss is computed over (batch size). The model is trained with Tensorflow [*Abadi et al.*, 2015] using the Adam optimizer [*Kingma and Ba*, 2015]. As metric we use the dice similarity coefficient (C_{Dice}) [*Dice*, 1945],

$$C_{Dice}(g,\hat{g}) = 2\frac{g \cap \hat{g}}{g+\hat{g}}. \tag{5.3.2}$$

Before computing the C_{Dice}, we set the values in the label and output that are above 0.1 to 1 and the rest to 0.

A first model, Model 1, is trained from scratch using the large synthetic data set that covers all possible source locations. During training we augment the synthetic data with different techniques such as random bulk time shifts, trace-varying gaussian noise, field noise and muting of individual traces (also known as station dropout [*Kriegerowski et al.*, 2019]). As for the field data, the synthetic data are also normalized by the maximum absolute amplitude during training. Next, a second model, Model 2, is trained by fine-tuning Model 1 on a much smaller field data set that consists of high magnitude events. Fig. 5.3 summarizes the two steps. The role of the synthetic data set is to provide a way to train a DNN from scratch when the amount of field data available is too small in the first place. Additionally, the synthetic data set covers all possible source locations and, therefore, Model 1, learns to associate seismic waveforms to all possible source locations. However, due to unaccounted differences between synthetic and real data, Model 1 is not capable of returning good source locations for lower magnitude events and instead only returns confident source locations for higher magnitude events that tend to have higher signal-to-noise ratios. Thus, in the second step, Model 1 is fined tuned using higher magnitude events. During this second training step the DNN learns to better recognize and extract relevant features in the field data, which allows it to better localize weaker magnitude events.
We experimented with freezing some of the weights, such as the entire encoder or decoder, during the fine tuning step. However, we did not observe any significant differences in the results. Therefore, we allowed all of the weights from Model 1 to change during the fine tuning step.

5.4 Results

As described in the previous section, we first train the DNN from scratch using the synthetic data set. The synthetically trained model, Model 1, was trained for 20 epochs with 51200 training samples, a batch size of 20 and a learning rate of 0.001. Training the model for a greater number of epochs did not significantly improve performance. Model 1 was then fine tuned using higher magnitude field

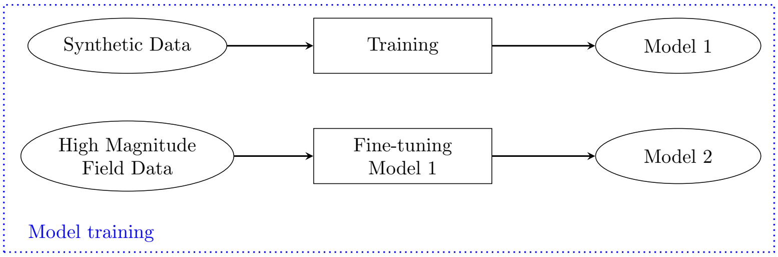


Figure 5.3: *Training DNN with synthetic data from scratch to get Model 1 and fine-tuning Model 1 using high magnitude field data to get Model 2.*

data events for training. Fig. 5.4 shows the magnitude distribution of the field data set (training/validation and test set).

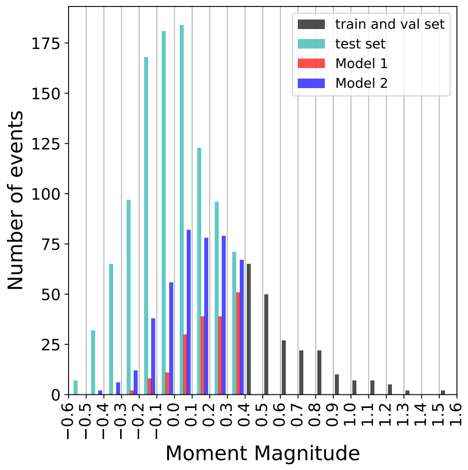


Figure 5.4: *Magnitude distribution of training set (black), test set (turquoise) and magnitude distributions of events passing peak-value threshold of 0.5 given Model 1 (red) and Model 2 (blue).*

We chose the events with moment magnitudes greater than 0.4 to create a field training and validation set with an 80/20% split. The field training set only contains 176 events and the validation set 44 events, whereas the test set contains 1025 events. The magnitude distribution of the training/validation and test set is shown in Fig. 5.4. Next, we fine tune the weights of Model 1 using the field data training set. We use early stopping to prevent the model from overfitting by analyzing the convergence over the C_{Dice}. If the C_{Dice} on the validation set does not increase by more than 0.001 in the course of 10 epochs training is stopped. Using this early stopping criterion, Model 2 was trained for 26 epochs. The learning rate was set to 0.0001 and the batch size to 2.

The test set contains all field data events with magnitudes less than 0.4, see Fig. 5.4. We apply Model 1 and Model 2 to the test set. As described above, the DNN returns the source location in terms of a 3D Gaussian distribution, which can take values between 0 and 1. In situations where the DNN is not able to extract features from the input efficiently, the output may not resemble a Gaussian distribution or it may have a lower maximum value, referred here as peak value. Thus, a threshold for the peak value can be used to accept or ignore the output. Here we set the threshold at 0.5 to keep things simple, however, the threshold could be selected based on some further analysis.

The magnitude distribution of the events in the test set that passed the threshold for Model 1 and Model 2 are shown in Fig. 5.4. We can observe that Model 1 covers a smaller range of magnitudes and also less events. We see that for Model 2 roughly 2 or more times as many events passed the threshold in the moment magnitude range between -0.3 and 0.3 compared to Model 1. Overall 420 events in the test set pass the threshold in Model 2, whereas for Model 1 only 180 events passed the threshold. This can be explained by Model 2 having better feature extraction capacities, as can be observed by comparing feature maps of the ninth hidden layer from Model 1 and 2 given the same input, as seen in Fig. 5.5: On the bottom row (Model 2) features associated to seismic waveforms are better visible than on the top row (Model 1) as highlighted by white rectangles.

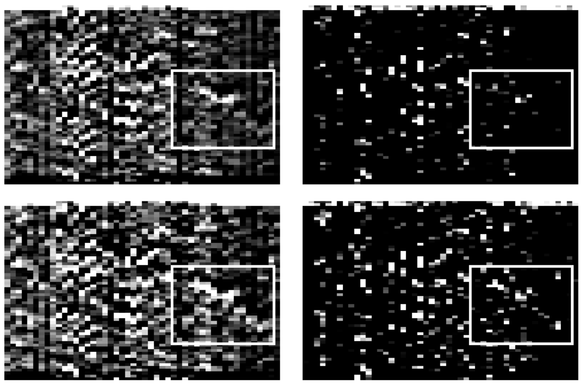


Figure 5.5: *Feature map outputs of the ninth hidden layer of Model 1 (first row) and Model 2 (second row) given same input. Part of waveform signal highlighted by white rectangles.*

Fig. 5.6 shows the epicenter locations predicted by Model 1 and Model 2 along with the DS-locations. In both situations we see an overall good match between the predicted locations and the DS-locations. However, we also see an improvement in location consistency in certain clusters in Model 2 compared to Model 1 (highlighted by round circles).

The hydraulic fracturing stimulations were undertaken at depths around 2000 to

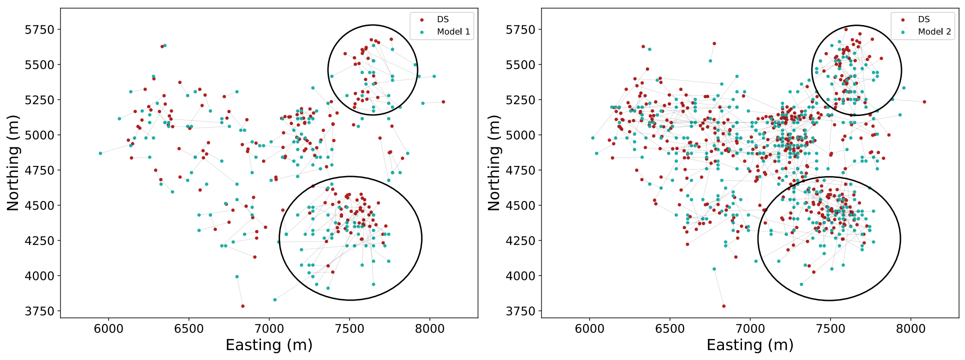


Figure 5.6: *Horizontal locations predicted by Model 1 (left) and Model 2 (right) passing threshold of 0.5 (turquoise dots) compared to locations computed by diffraction stacking (DS, red dots).*

2200 m. We plot the distributions of the predicted depths that passed the threshold given the test set for both models, as well as the DS computed depths that passed the threshold for Model 2, see Fig. 5.7. It is expected that the depths of the events concentrate and scatter around the hydraulic fracturing depth level. However, we observe that the DS computed depths scatter between 1900 and 2300 m. Model 1 more closely follows the expected depth distribution. However, Model 2 clearly follows the expected trend with a large number of events predicted at the hydraulic fracturing depth level. Thus, the fine tuning step also resulted in more consistent depth estimates compared to both the locations predicted by Model 1 and computed by DS.

Although fine tuning the model pretrained on synthetic data with higher-magnitude events helped localizing more low-magnitude events and also increased the localization accuracy, only about half the events in the test set passed the defined threshold. To increase the number of events passing the threshold the second model could be further refined by enlarging the training set. The larger training set could for instance include those events that passed the threshold for Model 2. We would then expect a similar observation as we have now: by fine tuning on more field data examples with lower magnitudes the updated model can extrapolate its localization capacities to events with lower quality data than that present in the training set.

The field training set used for fine tuning is very small. It may help to include a fraction of the synthetic data set in the fine tuning step to reduce the chance of overfitting due to the small training set.

For this case study we had a fairly good velocity model to generate the synthetic data. It would be interesting to investigate the importance the velocity model plays in generating synthetic data to pretrain the DNN. If pretraining with synthetic data that poorly resemble the field data does not affect the results after fine tuning on field data, the method could be employed to areas with poorly known velocity models.

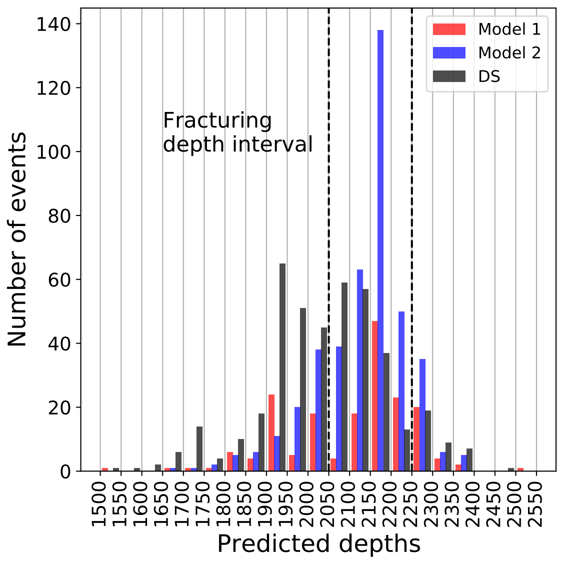


Figure 5.7: *Depth distribution predicted by Model 1 (red) and Model 2 (blue) passing threshold of 0.5.*

In this work only the vertical component of the seismic wavefield was used as input to obtain the source location, as this was the only component recorded in the field. The DNN-based source-localization methods presented in [*Kriegerowski et al.*, 2019; *Mousavi and Beroza*, 2020a; *Van den Ende and Ampuero*, 2020; *Zhang et al.*, 2020b] all made use of multiple component wavefields. It would be interesting to compare DNN models trained and applied to multiple component wavefields to those only using the vertical component.

Finally, we only considered situations where a single event is present in the input. Usually this is not a problem as in case multiple events are observed in the input they usually originate in proximity to each other. Still, to better understand what would happen if multiple events were present in the input, we simulate the presence of two events within a single time window under the same noise conditions and explore different situations: (1) same origin time and same magnitude/strength (Fig. 5.8a), (2) same origin time but different magnitudes (Fig. 5.8b), (3) different origin time but same magnitude/strength (Fig. 5.8c) and (4) different origin time and different magnitude (Fig. 5.8d). In the first situation Model 2 predicts a 3D distribution that lies somewhere in between both source locations. In the second situation one event is 10 times stronger than the other and Model 2 outputs a 3D Gaussian distribution at the stronger event's location. In the third situation where both events have the same magnitude but different origin time we observe a weak Gaussian distribution located nearby one of the event locations. Finally, if they have different origin times but one event dominates over the other, Model 2 predicts the location of the stronger event again. To summarize: the current network will choose the stronger event for localization and be located somewhere in between if the strength of the events is similar.

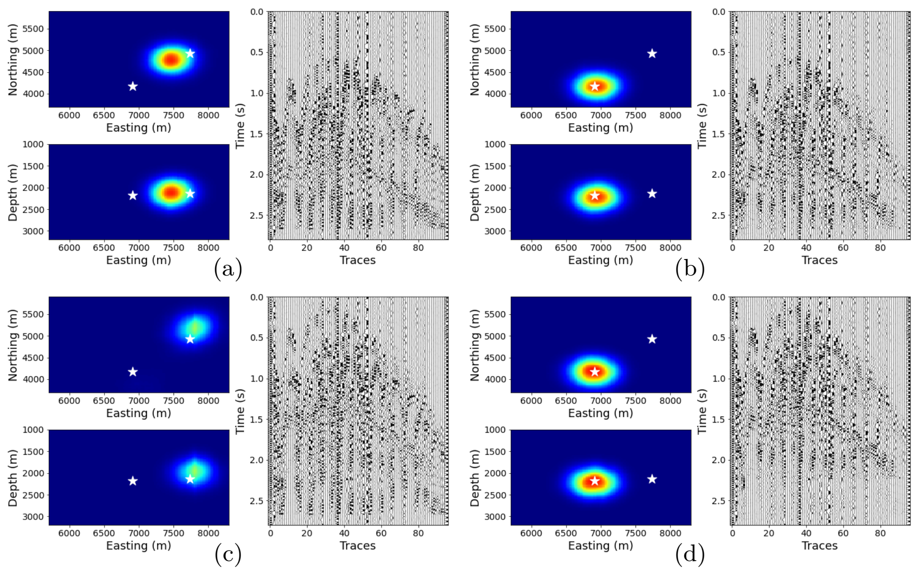


Figure 5.8: *Two synthetic events within same time-window including field noise. (a) same origin time and strength, (b) same origin time but different strength, (c) different origin time same strength and (d) different origin time and strength.*

Training Model 1 for 20 epochs takes roughly 13 hours on a Tesla T4 GPU, 140 seconds to train Model 2 for 26 epochs and loading and predicting a single input takes 0.01 seconds.

5.5 Conclusions

In this work we showed how a DNN can be trained on synthetic data to localize high magnitude field data events and map the location in terms of a 3D Gaussian distribution. The subsurface model used in this study was rather simple, which allowed the use of a computationally cheap modelling method to generate the synthetics. For complex subsurfaces other modelling algorithms that are computationally much more expensive would have to be used. This could pose a limitation on the number of synthetic training samples that can be generated.

The synthetically trained model was updated in a fine tuning step using a small number of high magnitude field data for training. This increased the capacity of the model to localize lower magnitude events. Not only does the fine-tuned model enable the localization of a larger number of events, but it also gives more consistent locations and better depth estimates compared to diffraction stacking.

6

Detecting microseismic events using a deep neural network trained for localization

"I've never seen a Lindy Hopper who wasn't smiling. It's a happy dance. It makes you feel good."

Frankie Manning

Abstract A crucial step to achieve real-time localization in microseismic monitoring is event detection. In previous chapters a deep learning algorithm (QNet) for microseismic localization was developed. It was shown that QNet generated a 3D Gaussian output with high maximum values representing the reconstructed source location, when presented with data containing microseismic events. From earlier work it was also observed that QNet's prediction provided a very faint distribution given noise in the input, for which the amplitude increased as soon as signal started to be present in the input. In this chapter we explore the possibility of using QNet for detection using three hours of continuous data from the same hydraulic fracturing site in Texas, USA. A simple detection method is developed that is fully based on the output generated by QNet. In the three hours of continuous data 14 events are detected by a diffraction stacking method. With the method developed in this chapter 12 of those events are detected, whilst two were missed. However, while there are some false detections, this method detects new events that were not detected by the diffraction stacking method.

6.1 Introduction

The very first step in the whole processing workflow of microseismic monitoring is event detection. There exist several signal detection methods in seismology one of the most popular being the short-term-long-term average (STA/LTA) [*Allen*, 1978]. The STA/LTA requires parameters defining the length of the short and long term window and a triggering threshold. Depending on how these parameters are set events may be missed or too many false detections may be triggered. Thus, these parameters have to be carefully selected. In microseismic monitoring the recorded signal associated with the operations commonly are around the same amplitude range as the noise or lower. Since the STA/LTA operates on individual time recorded signals it is not the most suitable detection algorithm for such situations.

To detect earthquakes with signals around or below the noise level and to distinguish them from other anthropogenic signals several other methods have been investigated. For instance the matched filter/template matching approach computes the crosscorrelation of the continuous data with a set of waveform templates [*Withers et al.*, 1998; *Shearer*, 1994]. However, this method can miss events for which there is no similar template. *Yoon et al.* [2015] presented a highly efficient detection algorithm (FAST) that is based on an algorithm originally designed to detect similar audio clips. FAST extracts features from the waveforms by creating a fingerprint, which is used to find similar pairs efficiently by using the local-sensitivity hashing method [*Andoni and Indyk*, 2006]. In microseismic monitoring the most commonly used detection method is based on diffraction stacking, and, thus also returns the location of the source. *Trojanowski and Eisner* [2017] provide a good overview and comparison of migration-based detection and localization methods.

In the past few years several machine learning and in particular deep learning based detection methods have been proposed. *Perol et al.* [2018] train a Convolutional Neural Network (CNN) on a single 3C seismic station for earthquake detection and classification into six large regional clusters. *Ross et al.* [2018a] trained a CNN to classify the input as containing either a P-wave, an S-wave or noise using a data set of millions of hand-picked arrivals. *Zhu et al.* [2019] developed a CNN based algorithm capable of learning to discriminate between noise and P- or S-phases using a much smaller data set with thousands instead of millions of examples. However, they mostly address signals with a high S/N. *Dokht et al.* [2019] used a CNN earthquake detector using the time-frequency domain representation of the seismograms as input. *Wu et al.* [2018b] take a very interesting approach to the detection problem as they identify the entire event window. This is particularly challenging because the duration can vary depending on the earthquake. To address this problem they developed a novel deep learning algorithm named Cascaded Region-based Densely Connected Network.

It is crucial that the detection methods are capable of distinguishing between waveforms associated to an earthquake and different impulsive noise sources. *Li et al.* [2018] address this issue by combining so-called generative adversarial networks (GANs) [*Goodfellow et al.*, 2014] and random forests [*Breiman*, 2001]. First, they train the GAN on P-wave data for it to essentially learn the key features in the

P-wave signal. Next, the random forest classifier is trained using P-wave and noise signals that were falsely triggered as events by an STA/LTA trigger. The signals are first fed to the feature extraction part in the GAN and transformed to a vector of features, which are passed to the random forest classifier. *Meier et al.* [2019] tested various different machine learning methods on a data set of local earthquakes with **M**>3. They found that the more complex models that directly extract features from the raw waveforms did better compared to the simpler models that were trained using user-defined features. *Mousavi et al.* [2019] developed a convolutional and recurrent neural network algorithm with the input data being represented as spectograms of the 3C seismograms. The model was trained using data recorded in Northern California. To test the generalization of their model they applied it to a different data set recorded in Guy-Greenbrier, Arkansas, where it reached a detection precision of 69%, which exceeded the precision of the FAST algorithm by 24%.

In chapters 3-5 of this thesis a deep learning method for the localization of microseismic events was presented (QNet). This method requires a multichannel input containing an event and maps it into a location output. Therefore, first an event must be detected and then be passed to QNet for localization. This detection step could be performed by several existing methods either based on classical signal processing or machine learning.

There are different possibilities to use QNet and turn it into a detection method. One possibility would be to keep the QNet architecture as is and turn QNet into a detection and localization network using noise and events as training data: for events in the input QNet would learn to reconstruct a 3D Gaussian distribution as before and for noise it would return zeros in the entire 3D space. Another way would be to adapt the architecture of QNet and train a separate network for the detection task only. Since detection is a binary task the entire decoder part of the QNet architecture is not needed. Thus, we could only take the encoder part of the network and add fully connected layers as well as a binary output layer, e.g. as shown in Fig. 6.1. Fig. 6.1 shows the original QNet architecture with the addition of fully connected layers at the bottom and where the grayed out decoder part would be removed. Furthermore, the training would not have to start from scratch since the convolutional layers were already trained for the localization problem and thus those layers were trained to extract features given microseismic events as input. Thus, only the weights in the newly added fully connected layers would have to be trained from scratch. Finally, a labeled training set would be needed with waveform and noise data. Since the weights in the convolutional layers are already trained, the training time is reduced and a smaller training set can be used. The trained detection model could next be used to differentiate between noise and signal in real time and whenever a signal is detected the data would be passed to the localization model to estimate the source location.

Another possibility would be to do something similar as *Kriegerowski et al.* [2019]: They trained a CNN for earthquake localization that returns the source coordinates (east, west and depth) and next investigated the possibility to use their model, trained purely on data containing events, for detection using the output of

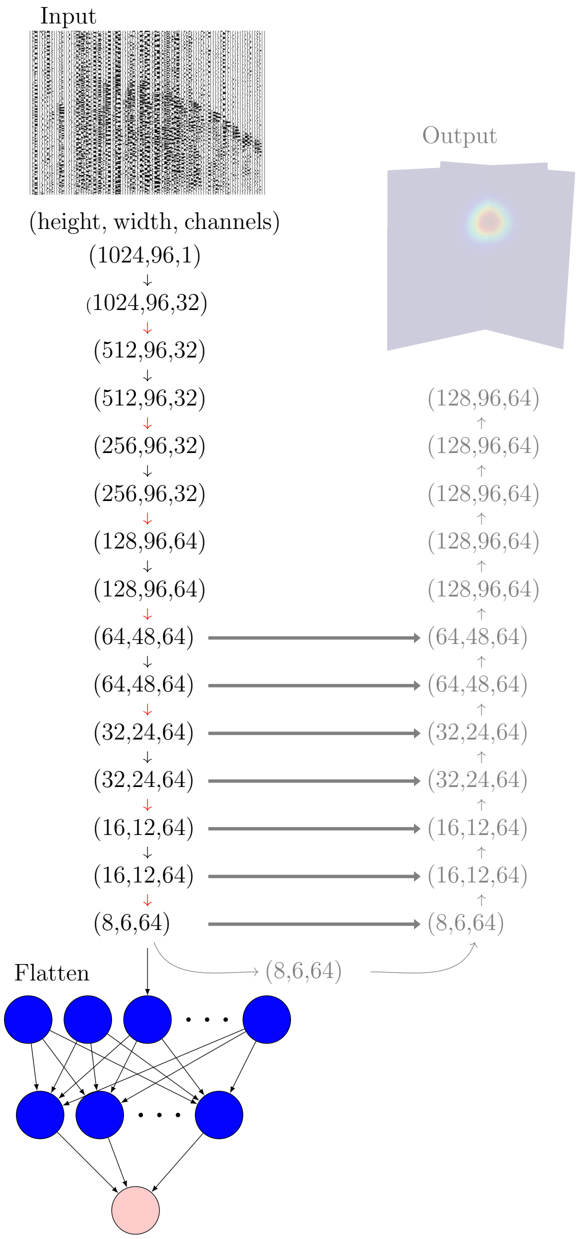


Figure 6.1: *Adapting original QNet architecture to use for transfer learning for event detection. All gray parts are dropped and fully connected layers are added at the bottom of the network to get a binary noise/signal classifier.*

the first layer of their localization model. Of the 400 detections they investigated (out of 6000) only 3.5% were false positives. Thus, it could be analysed how the output in the early convolutional layers could be used for detection. In previous chapters it was shown that QNet returned very low Gaussian outputs when presented with noise preceding an event. Thus, in this chapter the possibility of using QNet for detection is investigated, which would alleviate the need of using another detection method.

In the following the data, QNet and the detection method are described. This is followed by the results where two detection approaches are shown, as well as a breakdown analyzing events missed by the method and missed by diffraction stacking but detected by the method. Finally, this chapter is concluded with a discussion and conclusions.

6.2 Data, QNet and Method

■ 6.2.1 Data and QNet

QNet was trained on synthetic and field data recorded during ongoing operations at a hydraulic fracturing site in Texas, USA. Only time windows containing known events were used to train the model. In this chapter we analyze three hours of continuous data from the fifth day of monitoring recorded between 10 am and 1 pm. Thus, we use the QNet that was updated with field data up to the fourth day of monitoring (see Chapter 4). In those three hours of data a total of 14 events were detected by *Alexandrov et al.* [2020] (2 in the first, 8 in the second and 4 in the third hour).

■ 6.2.2 Method

In this chapter we investigate the possibility of directly using QNet by using the Gaussian output returned by QNet given seismic data as input. As we saw in the previous chapters the peak value of the Gaussian output can vary and tends to be lower for weaker signals. However, for most events, the maximum value in the output generated by QNet is high. Furthermore, it was observed that for time-windows only containing noise that precede an event QNet returns a weak output (Chapter 5). Thus, the output of the localization network could be used as an indicator for detection. In a way this would be similar to diffraction stacking which does both detection and localization.

Since for most events QNet returns a Gaussian distribution with a high value at its peak, a threshold on the maximum value in the output could be set to consider all of QNet's outputs with a maximum value above the threshold as a detection and the rest as noise. However, this could lead to missing events where the maximum value in the output is below the threshold and also to false detections where noise may fool QNet.

Since an event is present in multiple consecutive time windows the predicted output is likely going to be similar for those consecutive inputs. Thus, the maximum value in the predicted outputs is likely going to be similar as well and higher than the surrounding predictions. Thus, we can only consider those points that for n consecutive time windows are above a threshold. By doing so the threshold can be lowered, reducing the risk of missing an event with a lower maximum value in the prediction and at the same time reducing the number of false detections as random spikes are filtered out in the process. For convenience we refer to detections made with this method as QNetD and events detected by diffraction stacking as DS.

6.3 Results

As mentioned, QNet is applied continuously on three hours of data. QNet takes time windows of 2.8 seconds on 96 traces as inputs and the continuous data is fed to QNet every 0.4 seconds. This results in QNet processing 27000 time windows. Processing the three hours of data on a Intel(R) Xeon(R) CPU @ 2.20GHz, takes 2.75 hours, which involves loading 60 seconds of continuous data, removing all traces not used by QNet, band-pass filtering the data and feeding the data to QNet in time windows of 2.8 seconds.

■ 6.3.1 Detections

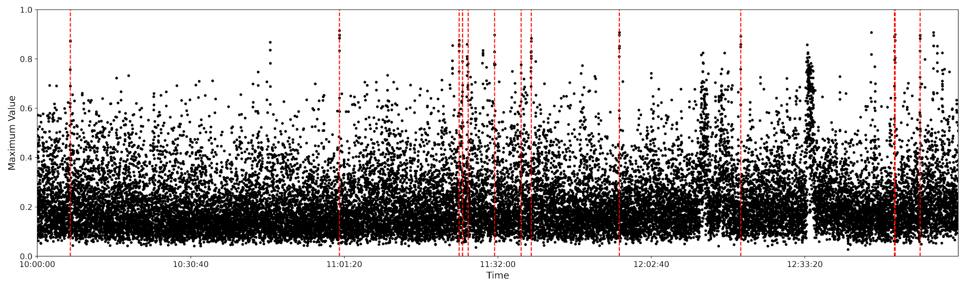
For each prediction the maximum value of the output and the location at the maximum output, as well as the time at the onset of the input is saved. Figure 6.2a shows the maximum value from QNet's output over all 27000 inputs. A large majority of those values are in the range between 0 and 0.4. We observe that for many of the events detected by diffraction stacking (red dashed vertical lines) the maximum value in the output predicted by QNet is higher with values around 0.7 and above. However, it is also observed that at similar regions with high maximum values, there are no events detected by diffraction stacking. Therefore, it is not straightforward to assess whether there is an event or not by looking at the maximum value given by the individual predictions.

By setting a threshold at 0.7 the number of predictions that potentially represent detections, is significantly reduced, see Fig. 6.2b. However, this leads to a high number of false positives and some events might still be missed if the maximum value in the prediction is below 0.7. Setting the threshold higher would potentially lead to more missed events and setting it lower to more false detections.

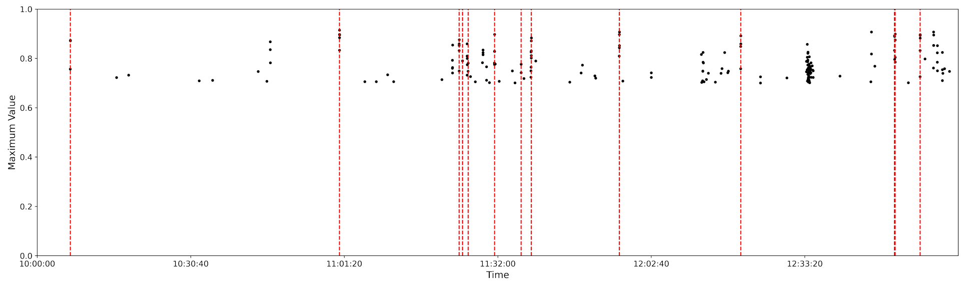
To increase the confidence that an event is truly detected, based on the maximum value in the predicted output, and at the same time lower the threshold, we only keep predictions where the maximum value is above a threshold of 0.5 for five consecutive times. As can be seen in Fig. 6.2c this results in a much smaller number of detections and at the same time most of the events given in the catalog are also detected.

■ 6.3.2 Normal Moveout Corrected Gathers

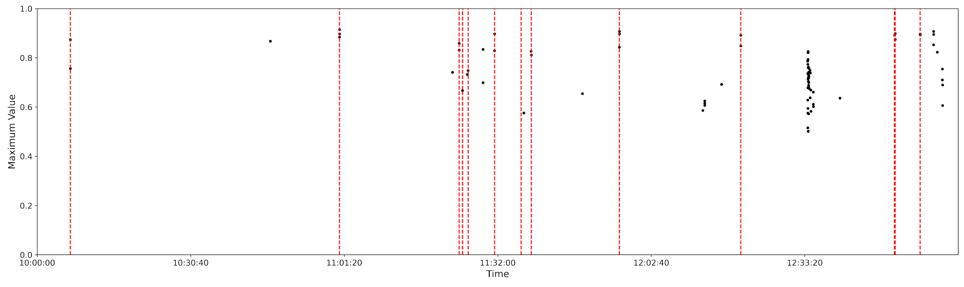
As can be seen in Fig. 6.2c QNetD results in more detections compared to diffraction stacking. While some of those detections might be due to there actually being an event, others might be false detections. Since the signal from weak events is around or below the noise level, it is difficult to determine whether an event is present or



(a)



(b)



(c)

Figure 6.2: *Maximum values of the predicted 3D-Gaussian output given continuous data as input: (a) Maximum value for each of the 27000 inputs over 3 hours of continuous data, (b) threshold of 0.7 applied to continuous output, (c) continuous output filtered by only keeping predictions with five consecutive maximum values above threshold of 0.5. Events detected by diffraction stacking are highlighted by red vertical lines.*

not just by looking at the input data. To better assess whether an event is present or not we compute the offset from the QNet predicted location to all receivers, i.e. not only the 96 receivers that are used for prediction. From those we keep the 200 closest receivers with respect to the predicted source locations. To do the NMO-correction, the traveltimes to each receiver are computed using an eikonal solver using the FTeikPy software [*Noble et al.*, 2014]. From the NMO-corrected gather it is easier to observe whether an event is present or not. However, the usefulness of the NMO-corrected gathers depends on how accurate the predicted locations are and how strong the signal is.

■ 6.3.3 Missed events

Out of the 14 events detected by diffraction stacking during that period of time, two events are missed by QNetD. For one of the two missed events the maximum value in the predicted output was never above 0.5 and instead ranged between 0.1 and 0.2. That event followed two seconds after the event detected at 11:23:06. The preceding event had a moment magnitude of 0.24 whereas the second event that was missed had a magnitude of -0.26. This weak magnitude event is difficult to observe even after applying a NMO-correction (Fig. 6.3c) using the layered P-velocity model and the source location determined by the diffraction stacking and relative location method [*Alexandrov et al.*, 2020]. The second missed event was not detected because the threshold of 0.5 was only reached three consecutive times instead of five.

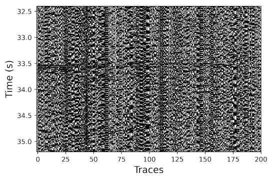
Table 6.1 summarizes the QNetD triggered detections and all DS detected events, as well as whether an event can be observed in the input and the mean maximum value over the five consecutive predictions around each detection.

■ 6.3.4 QNetD detections missed by diffraction stacking

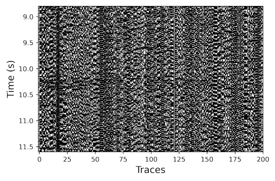
Between hour ten and eleven we have one detection that is not given in the diffraction stacking catalog at 10:45:33. It is surprising that this event is not given in the DS catalog as the event's signal can be clearly observed on the NMO-corrected gather, see Fig. 6.3a. The next QNetD detection not reported by DS can also be identified as being an event and was detected at 11:21:09. Although the signal strength is weaker compared to the event detected at 10:45:33, it can still be observed, see Fig. 6.3b. The next detection was triggered at 11:24:00. For this event it becomes more difficult to determine whether there is an event or not. A weak alignment can be identified, see Fig. 6.3d, but it is far less convincing than in the previous two cases. However, the NMO-corrected gather looks similar to the NMO-corrected gather of the event of magnitude -0.26 that was missed by QNetD, see Fig. 6.3c. We classified that detection as a "Maybe" in Table 6.1. The detection at 11:27:07 can be identified as a true detection, see Fig. 6.3e. For the detections at 11:36:52 and 12:55:53 it is not clear from the NMO-corrected gather if an event is observed or not, see Figs. 6.3f and 6.3i. Finally, the detection at 12:55:09 can be clearly classified as a true detection as the event is clearly visible, see Fig. 6.3h. Again, it is surprising that this event is not reported in the DS catalog.

■ 6.3.5 QNetD false detections

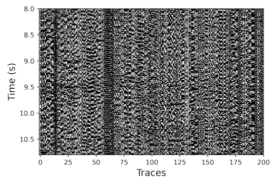
Between hour 11 and 12 QNetD gives two false detections. In the last hour QNetD returns many false detections. At 12:10 two false detections were triggered at second 4 and 26 and another false detection is determined at 12:13:43. Between 12:30:33 to 12:31:40 many detections were triggered. However, no events can be identified around this time. It is possible that during this time there was some correlated noise that was picked up by QNetD. However, this is difficult to determine only from the data. Finally, at 12:56:52 and 12:56:59 two more false detections were triggered.



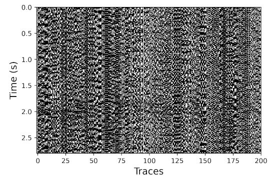
(a) 10:45:32



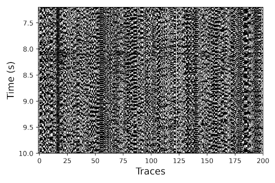
(b) 11:21:09



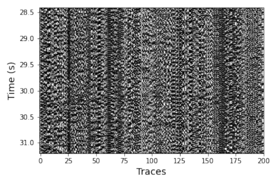
(c) 11:23:08



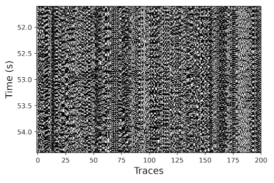
(d) 11:24:00



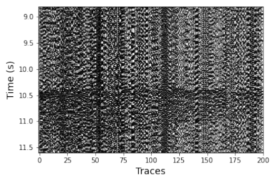
(e) 11:27:07



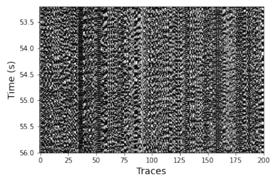
(f) 11:36:29



(g) 12:36:52



(h) 12:55:09



(i) 12:55:53

Figure 6.3: *NMO-corrected gathers of detections missed by QNetD (c), missed by DS but not by QNetD (a, b, e, h), detected by QNetD but not clear if event is present (d, f, g, i) .*

6.4 Discussion

QNetD was capable to detect events that were not in the diffraction stacking catalog. However, it also generated several false detections. In order to reduce these false detections the mean maximum value over the consecutive outputs that triggered a detection could help. We have seen that for many cases where an event was identified (either by DS or from the NMO-corrected gather) the mean maximum value is around 0.8 and for situations where there is no event this value is lower. However, this is not always the case, e.g. the missed event at 11:23:08 for which QNet returned low maximum values around 0.2. Furthermore, this could lead to wrongly classify detected events as false detections if their mean maximum value is below 0.8.

Another option could be to analyse how the locations predicted by QNet change

Table 6.1: *Summary of all DS detected events and QNetD detections.*

Detection time	DS	QNetD	Event	Mean max. value
10:06:28	Yes	Yes	Yes	0.83
10:45:32	No	Yes	Yes	0.83
10:59:04	Yes	Yes	Yes	0.9
11:21:09	No	Yes	Yes	0.75
11:22:27	Yes	Yes	Yes	0.85
11:23:06	Yes	Yes	Yes	0.67
11:23:08	Yes	No	Yes	0.19
11:24:00	No	Yes	Maybe	0.78
11:24:12	Yes	Yes	Yes	0.74
11:27:07	No	Yes	Yes	0.79
11:29:22	Yes	Yes	Yes	0.85
11:34:33	Yes	No	Yes	0.59
11:35:05	No	Yes	No	0.59
11:36:29	No	Yes	Maybe	0.77
11:36:33	Yes	Yes	Yes	0.83
11:46:32	No	Yes	No	0.67
11:53:45	Yes	Yes	Yes	0.89
12:10:04	No	Yes	No	0.62
12:10:26	No	Yes	No	0.62
12:13:43	No	Yes	No	0.68
12:17:29	Yes	Yes	Yes	0.87
12:30:33-12:31:40	No	Yes	No	0.52-0.72
12:36:52	No	Yes	Maybe	0.68
12:47:31	Yes	Yes	Yes	0.87
12:47:39	Yes	Yes	Yes	0.85
12:52:32	Yes	Yes	Yes	0.89
12:55:09	No	Yes	Yes	0.89
12:55:53	No	Yes	Maybe	0.82
12:56:52	No	Yes	No	0.72
12:56:54	No	Yes	No	0.66

from prediction to prediction. It is expected that if an event is present the predicted locations should be close to one another as long as the full event is present in the input. That information combined with QNetD could help reduce false detections. However, it would still not detect events that are missed due to a low Gaussian output returned by QNet.

It would also be possible to train a detector for example by using transfer learning and adapting the existing QNet architecture, as shown in Fig. 6.1, and combine that detection network with QNetD to increase the confidence in the detections.

6.5 Conclusions

We showed that a deep learning model trained for microseismic source localization can be used as a detector without additional training by making use of the predicted output of the localization model. While there were false detections, QNetD was able to detect events that were not detected by diffraction stacking. Furthermore, since the three hours of continuous data were processed in less than three hours the method shows real potential for real-time microseismic detection and localization using QNet. Finally, the application of QNet with QNetD resembles diffraction stacking, which also does both detection and localization in one go.

7 Conclusions and recommendations

"I don't need time, I need a deadline."

Duke Ellington

7.1 Conclusions

The increasing exploitation of natural resources close to urban areas have led to stronger regulations leading to a demand for real-time source detection and localization methods to comply with the regulations. The main aim of this thesis was to localize the source of induced microseismic events. In particular the aim was to address the localization of events at low signal-to-noise ratios. Additionally, a system that can be used for near real-time localization was sought. The preceding step of event localization is its detection. Thus, detection plays a crucial role in a real-time source-localization system. The detection step was only addressed in the last chapter of this thesis because the source-localization problem was the main driver of this research.

In chapter 2 a method requiring dense and regular spatial sampling of receivers over the monitoring site was investigated. This method involved the downward continuation of the recorded surface wavefields which focuses the signal at the time and location it originated from. It was shown that it is crucial to take the source mechanism into account to determine accurate hypocenters. Since the location and source mechanism are not known a priori, a set of filters were designed with each filter corresponding to a particular location and source mechanism in the area under investigation. By deconvolving the downward extrapolated wavefields with a set of filters the hypocenter and source mechanism can be determined by seeking the maximum amplitude. Good results were obtained for 2D synthetic data for both a simple subsurface model as well as the realistic Annerveen salt model and also when realistic noise was added.

In chapter 3 a deep learning algorithm based on a modified U-Net architecture is trained on synthetic data to be applied on field data. The need for the synthetic data was for one due to the small size of field data available to train a deep neural network from scratch and secondly, because of the problem that field data are rarely available before the start of underground industrial activities. The motivation behind the use of deep neural networks is that once trained the network can provide an output nearly instantly and at a low computational cost. Furthermore, deep neural networks have been proven to work extremely well at solving specific tasks. The synthetic data were generated using the same velocity model that was used to obtain the source locations using a diffraction stacking method. Random source locations within the defined source region as well as random double-couple source mechanisms were generated to model events. The addition of field noise to the synthetics proved to be an important step during training in order to obtain a robust network capable of accurately localizing events in field data. The source locations obtained using the deep neural network proved to be close to diffraction stacking as well as those refined by a relative location method for the events passing a given threshold. However, many of the lower magnitude events could not be accurately located by the network which was addressed in the subsequent chapter.

The localization of the lower magnitude events is addressed in chapter 4 by fine-tuning the synthetically trained deep learning algorithm using field data. First, a post-monitoring situation is addressed where all of the field data have been gathered and are next used in a transfer learning step to update the deep neural network (DNN) trained on synthetics. The updated DNN is far better compared to the previous DNN in localizing a much wider range of events. However, this post-monitoring application does not bring us closer to real-time source-localization. Therefore, a continuous acquisition mode workflow is proposed where the DNN is regularly updated using newly gathered field data. This further highlights the need of generating synthetic data to pretrain a DNN, since even less field data are initially available. The workflow is once applied only with the use of field data during the transfer learning step and once with both field and synthetic data.

The locations of the induced microseismic events migrate over time. Thus, there can be a discrepancy between the field data used for training compared to field data the DNN is applied to, which contain future events. DNNs are prone to overfit especially if trained on small data sets. Thus, the DNN updated only with field data tends to fail at accurately localizing the field data events. However, the DNN updated using both field and synthetic data provides more accurate locations because the synthetic data cover a wider range of locations and secondly, allows more data to be used during transfer learning, thus reducing the risk of overfitting.

In the discussion of chapter 4 it was mentioned that it may be possible to fine-tune the synthetically trained DNN using high magnitude field data in order to apply it on lower magnitude field data not present in the training set. This is explored in chapter 5. After fine-tuning the DNN trained purely on synthetic data using high magnitude field data the two DNNs are compared. First, a threshold was set based on the maximum value in the reconstructed 3D output by the DNNs to only keep predictions above the threshold. Compared to the DNN trained purely on synthetics the fine-tuned DNN returned better predictions and for a larger range of magnitudes and events. This shows that the DNN is able to extrapolate its feature extraction and source reconstruction capacities learned from higher $\mathbf{M}$ events to lower $\mathbf{M}$ events. By repeating the fine-tuning step several times each time including a larger range of $\mathbf{M}$ this trend may persist up to a certain point.

As mentioned at the start of these conclusions event detection is the first step towards an automated and near real-time event detection and localization workflow. In chapter 4 an example of an incoming event along with the predicted output showed how the localization DNN could be exploited for event detection. This is further addressed in chapter 6. Instead of training a new DNN from scratch or using parts of the localization-DNN for transfer learning to obtain a detection-DNN, the direct information returned by the localization-DNN is used to determine whether an event is detected in the input or not. This is achieved by considering a detection whenever the maximum value in the predicted output falls above a given threshold for a consecutive number of times. Since the localization-DNN was trained to extract features from the microseismic waveforms in the encoder and map those into a 3D location output it is assumed that whenever there are no waveforms in the input the output should not return a strong Gaussian distribution. It was shown that most of the events detected by diffraction stacking were also detected using this method. Furthermore, events not detected by diffraction stacking were detected. However, there were also some false detections where noise seems to have fooled the network. Nonetheless, this method enables real-time detection and localization on a single CPU. Finally, using the localization-DNN for both detection and localization somewhat resembles diffraction stacking methods as detection and localization are also performed together.

7.2 Recommendations

For this research several recommendations can be made for further investigation, which are listed in the following subsections.

■ 7.2.1 Data-driven earthquake localization and source-mechanism estimation based on wavefield extrapolation and 2D deconvolution in high-noise environments

The synthetic study in chapter 2 was only performed in 2D and thus, obvious next steps involve the extension to 3D and applying the method on field data. Technical challenges for the extension to 3D are the large volume of data gathered due to dense spatial sampling in both the inline and crossline directions and the high computational cost in forward modelling data for the creation of the filters as well as the downward continuation in 3D. Furthermore, the range of possible source mechanisms to consider significantly increases, even if only double-couple sources are considered due to the additional rake parameter. Advances in physics-informed neural networks *Raissi et al.* [2019] could be investigated for both forward modelling data through complex 3D models as well as for downward wavefield extrapolation. Another issue is that there is still the lack of microseismic monitoring systems with densely spatially sampled receivers. However, this may very well change in the near future as the industry's interest in distributed acoustic sensing continues to increase.

The detection problem was also not addressed in chapter 2. Detection could for instance be based on a threshold over the maximum value retrieved after downward continuation over all depths and deconvolving by all filters. However, this would be a costly detection scheme as it would involve the expensive downward continuation and deconvolution to be applied over all the data, which is not very practical, especially given that the majority of the recorded data contains noise. A cheaper alternative could be to downward continue the continuous data to a fixed depth level, thereby increasing the signal-to-noise ratio to some extent, and apply detection algorithms from this depth level.

■ 7.2.2 Investigate velocity model uncertainty

To generate the synthetic data to train the DNN, the same velocity model used for diffraction stacking was used. To take into account inaccuracies in the velocity model one might choose to use a set of different velocity models to generate the synthetic data. A DNN trained with those synthetic data may thereby be able to better handle uncertainties in the velocity model. It would also be of value to better understand the amount of data needed to train a DNN with synthetic data. Are more data needed to further improve the DNN or can a similar accuracy be obtained with less data?

■ 7.2.3 Investigate what makes a good synthetic training set

In general a good training set for a supervised machine learning algorithm is training data that is similar to the data the algorithm will be applied on. In this thesis, the synthetic data was generated using the same velocity model that was used to compute the diffraction stacking locations. The diffraction stacking locations were also used to create the labels for transfer learning. However, it would be valuable to investigate in depth what defines and how to evaluate a good synthetic training set.

■ 7.2.4 Train a DNN returning the posterior source location

In this thesis the source location is returned as a 3D Gaussian distribution with predefined standard deviation. However, this output has nothing to do with the posterior distribution or uncertainty distribution around the estimated source location. Therefore in the DNN used in this thesis the output gives little insight into location uncertainty, except partly by the magnitude of the output. The source location uncertainty is an important factor to take into account. Therefore, it would be interesting to include this information in deep learning applications. Bayesian neural networks [*MacKay*, 1992] and mixture density networks [*Bishop*, 1994] are possible candidates to investigate this topic.

7.2.5 Train a DNN to output source mechanisms

The strike, dip and rake angles of the double-couple components used to model the synthetic data were randomly sampled from a wide range of values to allow for a large variability in the data. However, if information about the expected source mechanisms is known a priori, it might be of value to model synthetics in a smaller range following the source mechanisms expected in the field. This may allow to train a DNN that is more targeted towards what is expected in the field data and therefore result in more accurate source locations. Finally, since the synthetic data are modelled and thus the source mechanism is known, the same synthetic data could be used to train a DNN to return the source mechanism.

7.2.6 Deep learning based localization

The DNN source localization method proposed in this thesis is tailored towards a specific monitoring site. It therefore can not be directly applied to a new monitoring site. For a new site new synthetics need to be computed taking into account the receiver locations, the velocity model and the expected source region at the new site. Furthermore, the entire DNN would have to be retrained from scratch. Perhaps it would be possible to start from a DNN trained at a different site but it would still require either field data or synthetics from that site. This is a drawback as generating synthetics and retraining a DNN are time-consuming tasks. This is much in contrast to other geophysical algorithms that were developed for imaging and inversion. Typical algorithms for source localization simply need the velocity model of the region and do not require it to be tuned depending on the site under investigation. I believe one of the main goals for future research in the field of artificial intelligence with applications in geophysics is the development of algorithms that can be generally applied. Developments in physics-informed neural networks definitely will become key to reach this goal. In any case artificial intelligence will continue to develop and I am curious to see where this journey will take us.

A

Supplemental Information to: Data-driven earthquake localization and source-mechanism estimation based on wavefield extrapolation and 2D deconvolution in high-noise environments

A.1 Basics of forward and inverse wavefield extrapolation

The acoustic wave equation in homogeneous media and without sources in the space-time domain is written as,

$$\nabla^2 p(\mathbf{x}, t) - \frac{1}{c^2}\frac{\partial^2 p(\mathbf{x}, t)}{\partial t^2} = 0, \tag{A.1.1}$$

where p is pressure, c the acoustic velocity, t is time, $\mathbf{x}$ is the position vector (x, y, z) and ∇^2 stands for the Laplace operator:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}. \tag{A.1.2}$$

Eq. A.1.1 can be transformed to the frequency-wavenumber domain and back again to the space-time domain using the temporal and spatial Fourier transform and the inverse transform, respectively. The temporal Fourier transform (FT) is defined as,

$$P(\mathbf{x}, \omega) = \int_{-\infty}^{\infty} p(\mathbf{x}, t) e^{-j\omega t} dt, \tag{A.1.3}$$

with ω representing the angular frequency and j the imaginary unit. The inverse of eq. A.1.3 is defined as:

$$p(\mathbf{x},t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} P(\mathbf{x},\omega)e^{j\omega t}d\omega. \tag{A.1.4}$$

Note that the temporal Fourier transform is described by capital letter. Similar equations exist for the transformation in the wavenumber domain, which are described with a tilde (~) above the letter. The spatial Fourier transform in x is defined as,

$$\tilde{p}(k_x,y,z,t) = \int_{-\infty}^{\infty} p(x,y,z,t)e^{jk_x x}dx, \tag{A.1.5}$$

where k_x is the horizontal wavenumber. Similarly its inverse is defined as:

$$p(x,y,z,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \tilde{p}(k_x,y,z,t)e^{-jk_x x}dk_x. \tag{A.1.6}$$

By applying the temporal FT, eq. A.1.3, and the spatial FT for both x and y, eq. A.1.5, to the acoustic wave equation, eq. A.1.1, we obtain the wave equation in the frequency-wavenumber domain:

$$\left(-k_x^2 - k_y^2 + \frac{\partial^2}{\partial z^2}\right)\tilde{P}(k_x,k_y,z,\omega) + \frac{\omega^2}{c^2}\tilde{P}(k_x,k_y,z,\omega) = 0. \tag{A.1.7}$$

Eq. A.1.7 can be written as,

$$\frac{\partial^2 \tilde{P}(k_x,k_y,z,\omega)}{\partial z^2} + k_z^2\tilde{P}(k_x,k_y,z,\omega) = 0 \tag{A.1.8}$$

with,

$$k_z = \begin{cases} \sqrt{\frac{\omega^2}{c^2} - k_x^2 - k_y^2} & \text{for } k_x^2 + k_y^2 \leq \frac{\omega^2}{c^2} \\ -j\sqrt{k_x^2 + k_y^2 - \frac{\omega^2}{c^2}} & \text{for } k_x^2 + k_y^2 > \frac{\omega^2}{c^2} \end{cases} \tag{A.1.9}$$

The solution of the two-way wave equation A.1.8 is given as,

$$\tilde{P}^+(k_x,k_y,z{=}\Delta z,\omega) = \tilde{P}^+(k_x,k_y,z{=}0,\omega)e^{-jk_z\Delta z} \tag{A.1.10a}$$
$$\tilde{P}^-(k_x,k_y,z{=}\Delta z,\omega) = \tilde{P}^-(k_x,k_y,z{=}0,\omega)e^{jk_z\Delta z}, \tag{A.1.10b}$$

where the + and - signs represent down- and up-going waves, respectively. For a detailed derivation of eqs. A.1.10a-A.1.10b the reader is referred to *Wapenaar and Berkhout* [1989] or *Gisolf and Verschuur* [2010]. The exponential in eq. A.1.10a and A.1.10b is known as the phase-shift or extrapolation operator $\tilde{W}$ in homogeneous media:

$$\tilde{W}(k_x,k_y,\omega,\Delta z) = \begin{cases} \tilde{W}^+ = e^{-jk_z\Delta z} \\ \tilde{W}^- = e^{jk_z\Delta z}. \end{cases} \tag{A.1.11}$$

Bibliography

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.

Afanasiev, M., C. Boehm, M. van Driel, L. Krischer, M. Rietmann, D. A. May, M. G. Knepley, and A. Fichtner (2019a), Modular and flexible spectral-element waveform modelling in two and three dimensions, *Geophysical Journal International*, *216*(3), 1675–1692, doi:10.1093/gji/ggy469.

Afanasiev, M., C. Boehm, M. van Driel, L. Krischer, M. Rietmann, D. A. May, M. G. Knepley, and A. Fichtner (2019b), Modular and flexible spectral-element waveform modelling in two and three dimensions, *Geophysical Journal International*, *216*(3), 1675–1692.

Alexandrov, D., L. Eisner, U. b. Waheed, S. I. E. Kaka, and S. A. Greenhalgh (2020), Normal faulting activated by hydraulic fracturing: A case study from the Barnett Shale, Fort Worth Basin, *The Leading Edge*, *39*(3), 204–211.

Allen, R. V. (1978), Automatic earthquake recognition and timing from single traces, *Bulletin of the seismological society of America*, *68*(5), 1521–1532.

Andoni, A., and P. Indyk (2006), Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, in *2006 47th annual IEEE symposium on foundations of computer science (FOCS'06)*, pp. 459–468, IEEE.

Anikiev, D., J. Valenta, F. Staněk, and L. Eisner (2014), Joint location and source mechanism inversion of microseismic events: Benchmarking on seismicity induced by hydraulic fracturing, *Geophysical Journal International*, *198*(1), 249–258.

Artman, B., I. Podladtchikov, and B. Witten (2010), Source location using time-reverse imaging, *Geophysical Prospecting*, *58*(5), 861–873.

Baysal, E., D. D. Kosloff, and J. W. Sherwood (1983), Reverse time migration, *Geophysics*, *48*(11), 1514–1524.

Bazargani, F., and R. Snieder (2013), Optimal wave focusing, *CWP Research Reports*, pp. 251–260.

Bishop, C. M. (1994), *Mixture density networks*, Aston University, Birmingham.

Bishop, C. M. (1995), *Neural networks for pattern recognitions*, Oxford University Press.

Breiman, L. (2001), Random forests, *Machine learning*, *45*(1), 5–32.

Chai, C., M. Maceira, H. J. Santos-Villalobos, S. V. Venkatakrishnan, M. Schoenball, W. Zhu, G. C. Beroza, C. Thurber, and E. C. Team (2020), Using a Deep Neural Network and Transfer Learning to Bridge Scales for Seismic Phase Picking, *Geophysical Research Letters*, *47*(16), e2020GL088,651.

Chambers, K., J.-M. Kendall, S. Brandsberg-Dahl, and J. Rueda (2010), Testing the ability of surface arrays to monitor microseismic activity, *Geophysical Prospecting*, *58*(5), 821–830.

Chambers, K., B. D. Dando, G. A. Jones, R. Velasco, and S. A. Wilson (2014), Moment tensor migration imaging, *Geophysical Prospecting*, *62*(4-Vertical Seismic Profiling and Microseismicity Frontiers), 879–896.

Chang, W.-F., and G. A. McMechan (1991), Wavefield extrapolation of body waves for 3-D imaging of earthquake sources, *Geophysical Journal International*, *106*(1), 85–98.

Chawla, N. V., N. Japkowicz, and A. Kotcz (2004), Special issue on learning from imbalanced data sets, *ACM SIGKDD explorations newsletter*, *6*(1), 1–6.

Chen, H., X. Meng, F. Niu, Y. Tang, C. Yin, and F. Wu (2018), Microseismic monitoring of stimulating shale gas reservoir in SW China: 2. Spatial clustering controlled by the preexisting faults and fractures, *Journal of Geophysical Research: Solid Earth*, *123*(2), 1659–1672.

Cunha, A., A. Pochet, H. Lopes, and M. Gattass (2020), Seismic fault detection in real data using transfer learning from a convolutional neural network pre-trained with synthetic seismic data, *Computers & Geosciences*, *135*, 104,344.

Deichmann, N., and D. Giardini (2009), Earthquakes induced by the stimulation of an enhanced geothermal system below Basel (Switzerland), *Seismological Research Letters*, *80*(5), 784–798.

Dice, L. R. (1945), Measures of the amount of ecologic association between species, *Ecology*, *26*(3), 297–302.

Dokht, R. M., H. Kao, R. Visser, and B. Smith (2019), Seismic event and phase detection using time-frequency representation and convolutional neural networks, *Seismological Research Letters*, *90*(2A), 481–490.

Duncan, P. M., and L. Eisner (2010), Reservoir characterization using surface microseismic monitoring, *Geophysics*, *75*(5), 139–146.

El Zini, J., Y. Rizk, and M. Awad (2019), A deep transfer learning framework for seismic data analysis: A case study on bright spot detection, *IEEE Transactions on Geoscience and Remote Sensing*, *58*(5), 3202–3212.

Ellsworth, W. L. (2013), Injection-induced earthquakes, *Science*, *341*(6142).

Fink, M. (1992), Time reversal of ultrasonic fields. I. Basic principles, *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, *39*(5), 555–566.

Foulger, G. R., M. P. Wilson, J. G. Gluyas, B. R. Julian, and R. J. Davies (2018), Global review of human-induced earthquakes, *Earth-Science Reviews*, *178*, 438–514.

Gajewski, D., and E. Tessmer (2005), Reverse modelling for seismic event characterization, *Geophysical Journal International*, *163*(1), 276–284.

Gaucher, E., M. Schoenball, O. Heidbach, A. Zang, P. A. Fokker, J.-D. van Wees, and T. Kohl (2015), Induced seismicity in geothermal reservoirs: A review of forecasting approaches, *Renewable and Sustainable Energy Reviews*, *52*, 1473–1490.

Gazdag, J. (1978), Wave equation migration with the phase-shift method, *Geophysics*, *43*(7), 1342–1351.

Gharti, H. N., V. Oye, M. Roth, and D. Kühn (2010), Automated microearthquake location using envelope stacking and robust global optimization, *Geophysics*, *75*(4), MA27–MA46.

Gisolf, D., and E. Verschuur (2010), *The principles of quantitative acoustical imaging*, EAGE Publications bv.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014), Generative adversarial nets, *Advances in neural information processing systems*, *27*.

Grechka, V., A. De La Pena, E. Schisselé-Rebel, E. Auger, and P.-F. Roux (2015), Relative location of microseismicity, *Geophysics*, *80*(6), WC1–WC9.

Gupta, H. K. (2002), A review of recent studies of triggered earthquakes by artificial water reservoirs with special emphasis on earthquakes in Koyna, India, *Earth-Science Reviews*, *58*(3-4), 279–310.

Hu, L.-Z., and G. A. McMechan (1988), Elastic finite-difference modelling and imaging for earthquake sources, *Geophysical Journal International*, *95*(2), 303–313.

Huang, L., J. Li, H. Hao, and X. Li (2018), Micro-seismic event detection and location in underground mines by using Convolutional Neural Networks (CNN) and deep learning, *Tunnelling and Underground Space Technology*, *81*, 265–276.

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

Johnson, S. W., D. J. Chambers, M. S. Boltz, and K. D. Koper (2021), Application of a convolutional neural network for seismic phase picking of mining-induced seismicity, *Geophysical Journal International*, *224*(1), 230–240.

Jozinović, D., A. Lomax, I. Štajduhar, and A. Michelini (2020), Rapid prediction of earthquake ground shaking intensity using raw waveform data and a convolutional neural network, *Geophysical Journal International*, *222*(2), 1379–1389.

Kao, H., R. Visser, B. Smith, and S. Venables (2018), Performance assessment of the induced seismicity traffic light protocol for northeastern British Columbia and western Alberta, *The Leading Edge*, *37*(2), 117–126.

Käufl, P., A. P. Valentine, R. W. de Wit, and J. Trampert (2016a), Solving probabilistic inverse problems rapidly with prior samples, *Geophysical Journal International*, *205*(3), 1710–1728.

Käufl, P., A. P. Valentine, and J. Trampert (2016b), Probabilistic point source inversion of strong-motion data in 3-D media using pattern recognition: A case study for the 2008 Mw 5.4 Chino Hills earthquake, *Geophysical Research Letters*, *43*(16), 8492–8498.

Kennett, B. L. (2005), ERZSOL3 (Reflectivity Method), `http://www.quest-itn.org/library/software/reflectivity-method.html`, accessed: 2020-03-30.

Kennett, B. L., and N. J. Kerry (1979), Seismic waves in a stratified half space, *Geophysical Journal of the Royal Astronomical Society*, *57*(3), 557–583, doi:10.1111/j.1365-246X.1979.tb06779.x.

Keranen, K. M., H. M. Savage, G. A. Abers, and E. S. Cochran (2013), Potentially induced earthquakes in Oklahoma, USA: Links between wastewater injection and the 2011 Mw 5.7 earthquake sequence, *Geology*, *41*(6), 699–702.

Keranen, K. M., M. Weingarten, G. A. Abers, B. A. Bekins, and S. Ge (2014), Sharp increase in central Oklahoma seismicity since 2008 induced by massive wastewater injection, *Science*, *345*(6195), 448–451.

Kingma, D. P., and J. L. Ba (2015), Adam: A method for stochastic optimization, *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1–15.

Kratz, M., A. Aulia, and A. Hill (2012), Identifying fault activation in shale reservoirs using microseismic monitoring during hydraulic stimulation: Source Mechanisms, b Values, and energy release rates, *CSEG Recorder*, *20*(06).

Kriegerowski, M., G. M. Petersen, H. Vasyura-Bathke, and M. Ohrnberger (2019), A deep convolutional neural network for localization of clustered earthquakes based on multistation full waveforms, *Seismological Research Letters*, *90*(2 A), 510–516, doi:10.1785/0220180320.

Li, H., Z. Xu, G. Taylor, C. Studer, and T. Goldstein (2017), Visualizing the loss landscape of neural nets, *arXiv preprint arXiv:1712.09913*.

Li, L., J. Tan, D. A. Wood, Z. Zhao, D. Becker, Q. Lyu, B. Shu, and H. Chen (2019a), A review of the current status of induced seismicity monitoring for hydraulic fracturing in unconventional tight oil and gas reservoirs, *Fuel*, *242*(January), 195–210, doi:10.1016/j.fuel.2019.01.026.

Li, L., J. Tan, B. Schwarz, F. Staněk, N. Poiata, P. Shi, L. Diekmann, L. Eisner, and D. Gajewski (2020), Recent advances and challenges of waveform-based seismic location methods at multiple scales, *Reviews of Geophysics*, *58*(1).

Li, M., H. Li, G. Tao, M. Ali, and Y. Guo (2019b), Microseismic event location using multi-scale time reversed imaging, *Journal of Petroleum Science and Engineering*, *174*, 144–160.

Li, Z., M.-A. Meier, E. Hauksson, Z. Zhan, and J. Andrews (2018), Machine learning seismic wave discrimination: Application to earthquake early warning, *Geophysical Research Letters*, *45*(10), 4773–4779.

Loewenthal, D., and I. R. Mufti (1983), Reversed time migration in spatial frequency domain, *Geophysics*, *48*(5), 627–635.

Lomax, A., A. Michelini, and D. Jozinović (2019), An investigation of rapid earthquake characterization using single-station waveforms and a convolutional neural network, *Seismological Research Letters*, *90*(2A), 517–529.

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

Ma, Y., S. Cao, J. W. Rector, and Z. Zhang (2020), Automated arrival-time picking using a pixel-level network, *Geophysics*, *85*(5), V415–V423.

MacKay, D. J. (1992), A practical bayesian framework for backpropagation networks, *Neural computation*, *4*(3), 448–472.

Majer, E. L., R. Baria, M. Stark, S. Oates, J. Bommer, B. Smith, and H. Asanuma (2007), Induced seismicity associated with enhanced geothermal systems, *Geothermics*, *36*(3), 185–222.

Maxwell, S. C., T. Urbancic, N. Steinsberger, R. Zinno, et al. (2002), Microseismic imaging of hydraulic fracture complexity in the Barnett shale, in *SPE annual technical conference and exhibition*, Society of Petroleum Engineers.

McGarr, A., D. Simpson, L. Seeber, and W. Lee (2002), Case histories of induced and triggered seismicity, *International Geophysics Series*, *81*(A), 647–664.

McKean, S., J. Priest, J. Dettmer, and D. Eaton (2019), Quantifying fracture networks inferred from microseismic point clouds by a Gaussian mixture model with physical constraints, *Geophysical Research Letters*, *46*(20), 11,008–11,017.

McMechan, G. A. (1982), Determination of source parameters by wavefield extrapolation, *Geophysical Journal International*, *71*(3), 613–628.

McMechan, G. A., J. Luetgert, and W. Mooney (1985), Imaging of earthquake sources in Long Valley caldera, California, 1983, *Bulletin of the Seismological Society of America*, *75*(4), 1005–1020.

McMechan, G. A., J. Wen, and J. A. Morales (1988), 3-D acoustic modelling and imaging for earthquake data, *Geophysical Journal International*, *92*(2), 339–344.

Meier, M.-A., Z. E. Ross, A. Ramachandran, A. Balakrishna, S. Nair, P. Kundzicz, Z. Li, J. Andrews, E. Hauksson, and Y. Yue (2019), Reliable real-time seismic signal/noise discrimination with machine learning, *Journal of Geophysical Research: Solid Earth*, *124*(1), 788–800.

Mendecki, A., G. Van Aswegen, and P. Mountfort (1999), A guide to routine seismic monitoring in mines, *A handbook on rock engineering practice for tabular hard rock mines*, p. 35.

Mendecki, A. J. (1993), Keynote address: Real time quantitative seismology in mines, *Rockbursts and Seismicity in Mines*, pp. 287–295.

Mousavi, S. M., and G. C. Beroza (2020a), Bayesian-Deep-Learning Estimation of Earthquake Location From Single-Station Observations, *IEEE Transactions on Geoscience and Remote Sensing*, *58*(11), 8211–8224, doi:10.1109/TGRS.2020.2988770.

Mousavi, S. M., and G. C. Beroza (2020b), A machine-learning approach for earthquake magnitude estimation, *Geophysical Research Letters*, *47*(1), e2019GL085,976.

Mousavi, S. M., S. P. Horton, C. A. Langston, and B. Samei (2016), Seismic features and automatic discrimination of deep and shallow induced-microearthquakes using neural network and logistic regression, *Geophysical Journal International*, *207*(1), 29–46.

Mousavi, S. M., W. Zhu, Y. Sheng, and G. C. Beroza (2019), CRED: A deep residual network of convolutional and recurrent units for earthquake signal detection, *Scientific reports*, *9*(1), 1–14.

Münchmeyer, J., D. Bindi, U. Leser, and F. Tilmann (2021), The transformer earthquake alerting model: a new versatile approach to earthquake early warning, *Geophysical Journal International*, *225*(1), 646–656.

Nair, V., and G. E. Hinton (2010), Rectified linear units improve restricted Boltzmann machines, in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Nicol, A., R. Carne, M. Gerstenberger, and A. Christophersen (2011), Induced seismicity and its implications for CO2 storage risk, *Energy Procedia*, *4*, 3699–3706.

Noble, M., A. Gesret, and N. Belayouni (2014), Accurate 3-d finite difference computation of traveltimes in strongly heterogeneous media, *Geophysical Journal International*, *199*(3), 1572–1585.

Pan, S. J., and Q. Yang (2009), A survey on transfer learning, *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345–1359.

Pearson, C. (1981), The relationship between microseismicity and high pore pressures during hydraulic stimulation experiments in low permeability granitic rocks, *Journal of Geophysical Research: Solid Earth*, *86*(B9), 7855–7864.

Perol, T., M. Gharbi, and M. Denolle (2018), Convolutional neural network for earthquake detection and location, *Science Advances*, *4*(2), 2–10, doi:10.1126/sciadv.1700578.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2019), Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, *Journal of Computational Physics*, *378*, 686–707.

Rietbrock, A., and F. Scherbaum (1994), Acoustic imaging of earthquake sources from the Chalfant Valley, 1986, aftershock series, *Geophysical Journal International*, *119*(1), 260–268.

Rodriguez, I. V., M. Sacchi, and Y. J. Gu (2012), Simultaneous recovery of origin time, hypocentre location and seismic moment tensor using sparse representation theory, *Geophysical Journal International*, *188*(3), 1188–1202.

Rodríguez-Pradilla, G., and D. W. Eaton (2020), Automated Microseismic Processing and Integrated Interpretation of Induced Seismicity during a Multistage Hydraulic-Fracturing Stimulation, Alberta, Canada, *Bulletin of the Seismological Society of America*, *110*(5), 2018–2030.

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*, pp. 234–241, Springer.

Ross, Z. E., M.-A. Meier, E. Hauksson, and T. H. Heaton (2018a), Generalized seismic phase detection with deep learning, *Bulletin of the Seismological Society of America*, *108*(5A), 2894–2901.

Ross, Z. E., M.-A. Meier, and E. Hauksson (2018b), P wave arrival picking and first-motion polarity determination with deep learning, *Journal of Geophysical Research: Solid Earth*, *123*(6), 5120–5129.

Rutqvist, J., F. Cappa, A. P. Rinaldi, and M. Godano (2014), Modeling of induced seismicity and ground vibrations associated with geologic CO2 storage, and assessing their effects on surface structures and human perception, *International Journal of Greenhouse Gas Control*, *24*, 64–77.

Rutqvist, J., A. P. Rinaldi, F. Cappa, P. Jeanne, A. Mazzoldi, L. Urpi, Y. Guglielmi, and V. Vilarrasa (2016), Fault activation and induced seismicity in geological carbon storage–lessons learned from recent modeling studies, *Journal of Rock Mechanics and Geotechnical Engineering*, *8*(6), 789–804.

Schultz, R., R. J. Skoumal, M. R. Brudzinski, D. Eaton, B. Baptie, and W. Ellsworth (2020a), Hydraulic fracturing-induced seismicity, *Reviews of Geophysics*, *58*(3), e2019RG000,695.

Schultz, R., G. Beroza, W. Ellsworth, and J. Baker (2020b), Risk-Informed Recommendations for Managing Hydraulic Fracturing–Induced Seismicity via Traffic Light Protocols, *Bulletin of the Seismological Society of America*, *110*(5), 2411–2422.

Shearer, P. M. (1994), Global seismic event detection using a matched filter on long-period seismograms, *Journal of Geophysical Research: Solid Earth*, *99*(B7), 13,713–13,725.

Simpson, D. W., W. Leith, and C. Scholz (1988), Two types of reservoir-induced seismicity, *Bulletin of the Seismological Society of America*, *78*(6), 2025–2040.

Staněk, F., D. Anikiev, J. Valenta, and L. Eisner (2015), Semblance for microseismic event detection, *Geophysical Journal International*, *201*(3), 1362–1369.

Suckale, J. (2010), Moderate-to-large seismicity induced by hydrocarbon production, *The Leading Edge*, *29*(3), 310–319.

Thorbecke, J. W., K. Wapenaar, and G. Swinnen (2004), Design of one-way wavefield extrapolation operators, using smooth functions in WLSQ optimization, *Geophysics*, *69*(4), 1037–1045.

Trojanowski, J., and L. Eisner (2017), Comparison of migration-based location and detection methods for microseismic events, *Geophysical Prospecting*, *65*(1), 47–63, doi:10.1111/1365-2478.12366.

Van den Ende, M. P., and J.-P. Ampuero (2020), Automated seismic source characterization using deep graph neural networks, *Geophysical Research Letters*, *47*(17), e2020GL088,690.

Van Der Baan, M., D. Eaton, and M. Dusseault (2013), Microseismic monitoring developments in hydraulic fracture stimulation, in *Isrm international conference for effective and sustainable hydraulic fracturing*, OnePetro.

van Thienen-Visser, K., and J. Breunese (2015), Induced seismicity of the Groningen gas field: History and recent developments, *The Leading Edge*, *34*(6), 664–671.

Van Wees, J., L. Buijze, K. Van Thienen-Visser, M. Nepveu, B. Wassing, B. Orlic, and P. Fokker (2014), Geomechanics response and induced seismicity during gas field depletion in the Netherlands, *Geothermics*, *52*, 206–219.

Vinard, N., G. Drijkoningen, and D. Verschuur (2022), Localizing microseismic events on field data using a U-Net-based convolutional neural network trained on synthetic data, *Geophysics*, *87*(2), KS33–KS43.

Vlek, C. (2018), Induced earthquakes from long-term gas extraction in Groningen, the Netherlands: Statistical analysis and prognosis for acceptable-risk regulation, *Risk analysis*, *38*(7), 1455–1473.

Wang, Y., B. Wang, N. Tu, and J. Geng (2020), Seismic trace interpolation for irregularly spatial sampled data using convolutional autoencoder, *Geophysics*, *85*(2), V119–V130.

Wapenaar, C., and A. Berkhout (1985), Wave field extrapolation techniques for inhomogenous media which include critical angle events. Part I: Methods using the one-way wave equations, *Geophysical Prospecting*, *33*(8), 1138–1159.

Wapenaar, C. P. A., and A. J. Berkhout (1989), *Elastic wave field extrapolation: Redatuming of single-and multi-component seismic data*, vol. 2, Elsevier.

Whitmore, N. D. (1983), Iterative depth migration by backward time propagation, in *SEG Technical Program Expanded Abstracts 1983*, pp. 382–385, Society of Exploration Geophysicists.

Withers, M., R. Aster, C. Young, J. Beiriger, M. Harris, S. Moore, and J. Trujillo (1998), A comparison of select trigger algorithms for automated global seismic phase and event detection, *Bulletin of the Seismological Society of America*, *88*(1), 95–106.

Wu, B., A. Wan, X. Yue, P. Jin, S. Zhao, N. Golmant, A. Gholaminejad, J. Gonzalez, and K. Keutzer (2018a), Shift: A zero flop, zero parameter alternative to spatial convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9127–9135.

Wu, Y., Y. Lin, Z. Zhou, D. C. Bolton, J. Liu, and P. Johnson (2018b), DeepDetect: A cascaded region-based densely connected network for seismic event detection, *IEEE Transactions on Geoscience and Remote Sensing*, *57*(1), 62–75.

Xuan, R., and P. Sava (2010), Probabilistic microearthquake location for reservoir monitoring, *Geophysics*, *75*(3), MA9–MA26.

Yang, J., and H. Zhu (2019), Locating and monitoring microseismicity, hydraulic fracture and earthquake rupture using elastic time-reversal imaging, *Geophysical Journal International*, *216*(1), 726–744.

Yoon, C. E., O. O'Reilly, K. J. Bergen, and G. C. Beroza (2015), Earthquake detection through computationally efficient similarity search, *Science advances*, *1*(11), e1501,057.

Zhang, G., C. Lin, and Y. Chen (2020a), Convolutional neural networks for microseismic waveform classification and arrival picking, *Geophysics*, *85*(4), WA227–WA240.

Zhang, X., J. Zhang, C. Yuan, S. Liu, Z. Chen, and W. Li (2020b), Locating induced earthquakes with a network of seismic stations in Oklahoma via a deep learning method, *Scientific Reports*, *10*(1), 1–14, doi:10.1038/s41598-020-58908-5.

Zhebel, O., and L. Eisner (2015), Simultaneous microseismic event localization and source mechanism determination, *Geophysics*, *80*(1), KS1–KS9.

Zhou, Y., H. Yue, Q. Kong, and S. Zhou (2019), Hybrid event detection and phase-picking algorithm using convolutional and recurrent neural networks, *Seismological Research Letters*, *90*(3), 1079–1087.

Zhu, L., Z. Peng, J. McClellan, C. Li, D. Yao, Z. Li, and L. Fang (2019), Deep learning for seismic phase detection and picking in the aftershock zone of 2008 Mw 7.9 Wenchuan Earthquake, *Physics of the Earth and Planetary Interiors*, *293*, 106,261.

Zhu, W., and G. C. Beroza (2019), PhaseNet: a deep-neural-network-based seismic arrival-time picking method, *Geophysical Journal International*, *216*(1).

Zoback, M. D., and S. M. Gorelick (2012), Earthquake triggering and large-scale geologic storage of carbon dioxide, *Proceedings of the National Academy of Sciences*, *109*(26), 10,164–10,168.

Acknowledgments

The first time I talked to Guy and Eric was over Skype during the interview that led to this work. I had a very good feeling about both of them and about the research topic. So when a few days later Guy and Eric gave me the opportunity to meet them in person in Delft, I gladly accepted. The in person conversations were even better and I went back home with a really good feeling and accepted the position.

During my PhD in Delft I felt really lucky to have two supervisors, which both made it a priority to meet once a week to discuss progress and exchange ideas. Without their continuous support none of this would have been possible.

A couple of months after joining TU Delft I had the opportunity to take a 3-day course arranged by the EAGE and presented by Leo Eisner. Meeting Leo was not only important for the learning experience during the course, but also later into my PhD journey when I was looking for a field dataset to test some of the methods presented in this thesis on field data. Leo was also interested in the research and contributed to chapter 4 of this thesis, thereby increasing its quality and giving it the needed industry perspective it was previously lacking.

During my time in Delft I had the opportunity to meet many wonderful PhD students from the research group such as my desk neighbor Gil Averbuch whom I thank for the many great talks over coffee and beer. Also many thanks to Max Holicki which always made us remember it was time for lunch. A big thank you to all that joined for lunch, which pretty much always led to absurd and hilarious conversations with Chris, Reuben, Joeri, Billy, Johno, Florencia, Jan Willem, Diego and Musab. Chris, thank you for offering me a homey place to sleep when I came back to the Netherlands for a couple of weeks last summer, you are awesome and I wish you all the best. Musab, thank you for all the conversations we had for the better part of two years in Delft and remotely thereafter. I wish you all the best and I hope we keep in touch.

As a coffee lover I thank all the baristas who served me tasty coffee at any point during these last years and a special thanks to Café Labs, who brought new life to

TU Delft.

I also thank all of the Lindy Hoppers in the Netherlands that I had the pleasure to dance with. Lindy Hop, you contributed so much to this and helped me keep sane when I was down. Through Lindy Hop I also met Daniel Fan another fellow Lindy Hop dancer and TU Delft researcher who became a good friend.

During my first few months in Delft I ran into one of my neighbors, Alphonse. We had a short chat which led to a few beers and ever since we became friends. Thank you for all the great talks about life over dinner and beer.

I also want to thank Urs, Marko and Erben my good friends that also came to visit me in the Netherlands on separate occasions for all the wonderful and fun times.

None of this would have been possible without my family. Thank you mum, dad and Sabrina for always being there for me and all your love.

Being in a long distance relationship does not make the PhD journey easier so I want to first thank Guy and Eric once more for being so kind and understanding as to give me the flexibility to work from Zurich from time to time. Being able to see Sandra more frequently truly helped me stay alive throughout this journey. Sandra, thank you for everything. I would not have been able to write this thesis without you being there.

Nicolas Vinard,
Zurich,
May 2022

Curriculum Vitæ

Nicolas André Vinard

23-04-1989	Born in Lausanne, Switzerland.

Education

2005–2009	Gymnasium Kantonsschule Baden, Switzerland.
2010–2014	B.Sc. Earth Sciences Swiss Federal Institute of Technology, ETH Zurich.
2014–2017	M.Sc. Earth Sciences Swiss Federal Institute of Technology, ETH Zurich. *Thesis:* Optimal Design in Ultrasound Tomography for Breast Cancer Detection *Promotor:* Prof. A. Fichtner

Professional Experience

2017	Research assistant at the Swiss Federal Institute of Technology Institute of Geophysics, Seismology and Wave Physics lab

List of publications

■ Journal Publications

[3] **Vinard, N. A.**, Drijkoningen, G. G. and Verschuur, D. J. (2022), *Localizing microseismic events on field data using a U-Net based convolutional neural network trained on synthetic data*, Geophysics, 87, 33–44. Chapter 3 of this thesis

[2] **Vinard, N. A.**, Drijkoningen, G., G., Verschuur, D., J., Alexandrov, D. and Eisner, L. (2022), *Localizing weak microseismic events using transfer learning with a deep neural network*, Geophysical Prospecting. Chapter 4 of this thesis

[1] **Vinard, N. A.**, Drijkoningen, G. G. and Verschuur, D. J. (2021) *Fine tuning a deep neural network to localize low magnitude earthquakes*, 2021 International Conference on Electrical, Computer and Energy Technologies (ICECET), pp. 1-6., IEEE Chapter 5 of this thesis

Note that minor changes have been introduced to make the text consistent with the other chapters.

■ Conference abstracts

[3] **Vinard, N. A.**, Drijkoningen, G. G. and Verschuur, D. J. (2021), *Towards synthetically trained Convolutional Neural Networks for induced earthquake detection and localization*, EAGE GeoTech 2021 First EAGE Workshop on Induced Seismicity.

[2] **Vinard, N. A.**, Drijkoningen, G. G. and Verschuur, D. J. (2020), *Real-data earthquake localization using convolutional neural networks trained with synthetic data*, SEG Technical Program Expanded Abstracts, pp. 1576–1580.

[1] **Vinard, N. A.**, Drijkoningen, G. G. and Verschuur, D. J. (2019), *Data-driven earthquake detection, localization and source mechanism estimation based on wavefield extrapolation and 2D deconvolution in high noise environments*, 2019 Annual Meeting : Seismological Society of America Technical Sessions. pp. 963–964. Part of Chapter 2 of this thesis