UNIVERSITY OF AARHUS

MASTER THESIS

Machine Learning Based Detection of Couplings in tTEM Data

Author:
Toke FREDERIKSEN
201406925

Supervisor:
Lektor. Jakob Juul LARSEN

Geophysics, Instrumentation and Signal Processing Group
Department of Engineering
Section of Electrical and Computer Engineering

July 11, 2025

UNIVERSITY OF AARHUS

Abstract

Faculty of Science and Technology
Department of Engineering
Section of Electrical and Computer Engineering

Machine Learning Based Detection of Couplings in tTEM Data

Toke FREDERIKSEN
201406925

The transient electromagnetic method is a well established geophysical method for delineation of the subsurface. It is commonly used in mineral exploration, groundwater exploration or environmental mapping among other use cases. Recently, a new instrument for high-resolution mapping of resistivity structures in the shallow subsurface has been developed. As with all transient electromagnetic methods, the system is sensitive to electromagnetic coupling to man-made conductors. A significant amount of total data processing time is spent identifying and culling couplings. The thesis aims to identify a machine learning based method for efficient automatic detection of couplings. The problem was approached starting from a simple random forest model and from there identifying underlying limitations and problems leading to a more advanced model. In conclusion, an unsupervised one-class autoencoder neural network were developed and achieved $\sim$ 90% hit rate, paving the way for fully automated detection of couplings by use of synthetic data.

Acknowledgements

I would like to acknowledge the people who have enabled me to write this thesis. Firstly, thank you to Jakob Juul Larsen who always provided excellent guidance and showed genuine interest in my work, after a discussion with Jakob my morale was always high. Thank you to the people at the HGG group for providing data and helpful discussions. Lastly, a thank you to friends and family for support and constructive comments. Thank you to my friend Peter Granum for proof reading and lastly a special thank you to my various office mates, who made the long working days better.

Contents

List of Figures

List of Tables

List of Abbreviations

AUC	**A**rea **U**nder **C**urve. Specifically the area under the ROC curve.
CNN	**C**onvolutional **N**eural **N**etwork.
FPR	**F**alse **P**ositive **R**ate.
MAE	**M**ean **A**bsolute **E**rror.
ReLU	**Re**ctified **L**inear **U**nit.
ROC	**R**eceiver **O**perating **C**haracteristics.
(t)TEM	(**t**owed) **T**ransient **E**lectromagnetic **M**ethod.
TPR	**T**rue **P**ositive **R**ate.

Chapter 1

Introduction

The transient electromagnetic method (TEM) is a well-established geophysical method for delineation of subsurface resistivity of the earth [1]. The method is commonly employed for mineral exploration, groundwater exploration, and for environmental mapping e.g. [2], [3]. The method relies on inducing an electromagnetic field in the subsurface and measuring the decay response of the induced field. Recently, a new instrument for high-resolution mapping of resistivity structures in the shallow surface of the earth has been developed in the Hydrogeophysics Group at Aarhus University. The new instrument, denoted tTEM for towed Transient Electromagnetic Method, is based on technology known from the airborne SkyTEM system [4].

A particular attractive feature of airborne systems is that they can cover vast areas in a short time due to their high speeds, often well in excess of 100 km/h. However, one drawback of airborne systems is the altitude of the TEM transmitter and receiver coils, that leads to a significant footprint and a corresponding low lateral resolution. Further, the separation between flight lines is typically several hundreds of meters presenting another barrier for high lateral resolution. For high-resolution 3D mapping of resistivity structures in the shallow subsurface, towed ground-based methods can therefore be preferred

In contrast to the airborne SkyTEM system, which is carried by a helicopter, the tTEM system is groundbased. An ATV drags two frames with a transmitter and receiver coil. Two advantages of the tTEM setup are a much smaller footprint and a negligible distance between the coils. The system has delivered impressive results and hitherto unknown resistivity structures on a sub-10 m scale can be mapped with the system.

Any TEM measurement is sensitive to electromagnetic coupling to manmade structures, e.g. powerlines, burried wires and other electrical conductive structures. Couplings distort the TEM signals and data affected by couplings must be removed from the acquired TEM data sets before inversion. Inversion implies fitting model parameters to observations to reach the minimum misfit between measured and computed electromagnetic fields. The best fit parameters make up a layered model of the earth, described by layer depth and resistivity.

Today, rejection of couplings is an expensive, labor-intensive process and the goal of the project is to investigate machine learning based methods for automated detection of couplings in tTEM data. Efficient solutions to this problem are fast becoming a necessity as the tTEM system is increasingly being used e.g. in the new Innovation Fund Denmark sponsored MapField project. Prior work on this topic has achieved $\sim$90% success rates in recognizing coupled data. However, such rates are too low to be used for production. One limitation in achieving higher success rates is poorly labelled training data and solutions to this problem will be a an area of focus in the project. Another important aspect is to achieve models that generalize from one geological setting to another.

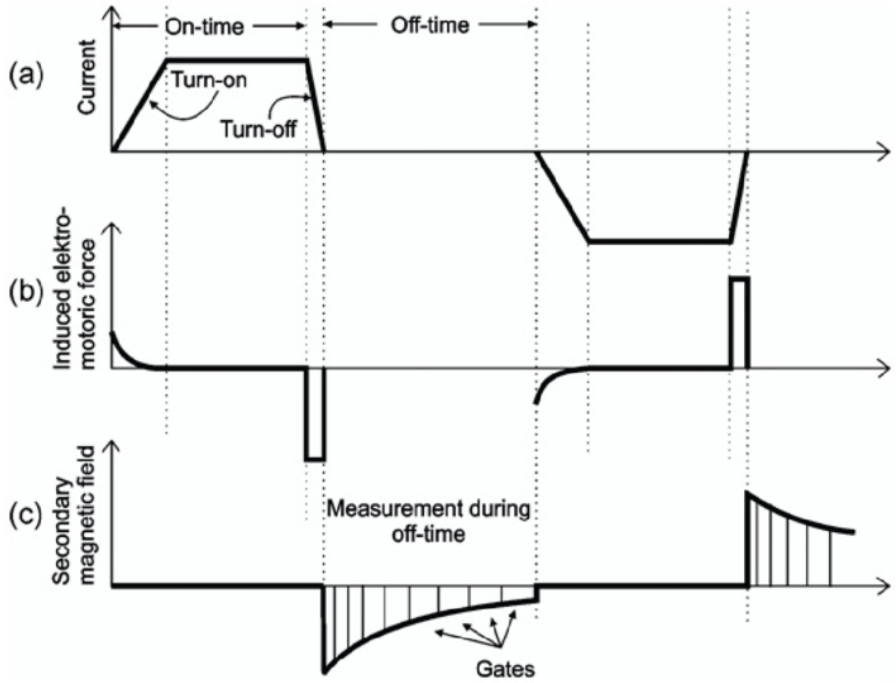


FIGURE 1.1: Basic nomenclature and principles of the TEM method; (a) shows the current in the transmitter loop; (b) is the induced electromotive force in the ground; and (c) is the secondary magnetic field measured in the receiver coil. Adapted from [5].

1.1 Transient Electromagnetic Method

The basic principle of the transient electromagnetic method consist of inducing an electromagnetic field in the earth and subsequently measuring the decay response of the induced field. A primary magnetic field is created by passing a strong transient current through a transmitter coil. When the current is abruptly turned off, the changing magnetic field induce eddy currents in the surrounding conductive earth by Faraday's law of induction. In turn the induced eddy currents give rise to a secondary magnetic field. Due to Ohmic losses in the ground, the secondary magnetic field decay over time. The decay response dB/dt is measured by a receiver coil. An overview of the process is found in figure 1.1.

The TEM measurements used in this thesis have been conducted using the tTEM system [3], illustrated in figure 1.2. tTEM is a ground based system capable of imaging the subsurface with a depth of investigation up to 70 m, with a distance between soundings of 3 to 4 meters. The configuration consist of a transmitter- and receiver coil, towed behind an all terrain vehicle (ATV). The system is driven in straight lines across the survey areas, while continuously performing measurements. Several transients measured in the receiver coil are stacked together (averaged), into what is commonly known as a sounding. A sounding is divided in logarithmic increasing time windows to improve signal to noise ratio at later times, a technique known as log-gating illustrated in figure 1.1. The time to record a sounding is about 0.7 seconds.

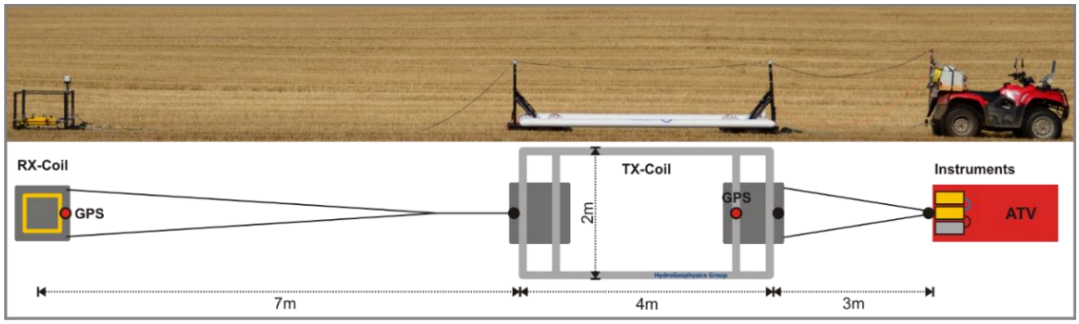


FIGURE 1.2: The tTEM system, Tx is the transmitter coil and Rx the receiver coil. Adopted from [3]

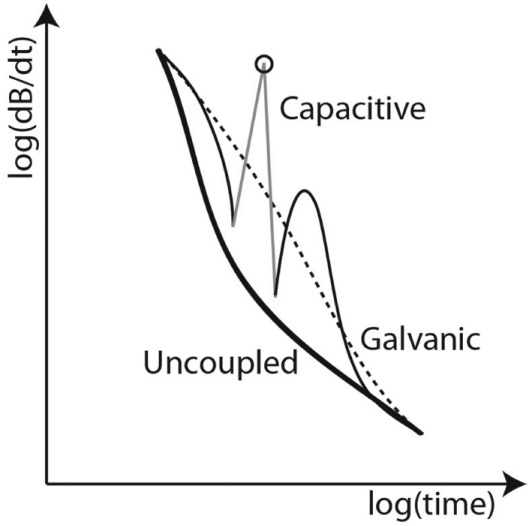


FIGURE 1.3: A schematic plot of an uncoupled sounding (bold line), a capacitive coupled sounding (thin line), and galvanic coupled sounding (thin dashed line) on a log-log scale. The capacitive sounding changes sign, indicated by grey connecting line. The negative point itself is shown with a circle. Adopted from [6].

1.1.1 Couplings

The tTEM system and TEM measurements in general are very sensitive to electromagnetic couplings to man-made structures such as power lines, cables and other electrical conductors. The goal of the thesis is to design a machine learning based model to automatically cull coupled data. The couplings can be divided in two different types; galvanic and capacitive as illustrated in figure 1.3. The illustration show soundings as gate values as a function of gate time. We will primarily work with soundings in a time series format ie. several dB/dt values at each measurement time, an example is found in figure 2.5.

Galvanic and capacitive couplings appear different as they arise from different phenomena and are not necessarily distinguishable directly from the data. The capacitive coupling can be modelled as an oscillatory, exponentially decaying LCR (inductor, capacitor and resistor) circuit. A source for such a coupling could be an underground wire, with the wire and insulation acting as capacitor, the earth as resistive current return path and the whole circuit as one large inductor. When a current is induced in the circuit, the LC components will give rise to an oscillation

with exponential damping due to the resistive earth. The galvanic coupling can be modelled as an LR circuit. An overhead grounded power line could be as a source for galvanic coupling, with two or more masts connected by the power line acting as LR in connection with the earth. This type of coupling can only be identified by comparing neighbouring soundings and identifying spatial variations deemed too abrupt to be caused by changes in earth resistivity [6]. In practice the galvanic coupling is therefore very difficult to distinguish from a legitimate measurement, as it only give rise to an exponential decay and no oscillatory signal like its capacitive counterpart. In this project we will therefore focus on detection of capacitive couplings, to narrow the scope of the thesis. Other types of noise may also occur such as induced polarization (IP) in a sufficiently polarizable earth [7]. However, we will ignore this type of noise as it is beyond the scope of the thesis.

The dB/dt voltage data obtained from tTEM measurements are divided in three categories according to their processing state: 1) instrument data, raw unstacked transients from the receiver; 2) raw data, stacks of transients (instrument data); and 3) averaged data, e.g. raw data stacks [8]. The second stage is the state of the data as used in this thesis. Before the data is averaged in step 3, couplings must first be removed. The dB/dt values are measured in units of volt per ampere per square meter and usually ranges in magnitude from 10^{-10} to 10^{-6}. All data used in this thesis have been collected and processed by the HGG group [9]. The data has been manually inspected and annotated such that coupled soundings have received a label of 0 and non coupled soundings a label of 1. These markings form the basis of the supervised learning in this project and as such represent a binary classification problem; decide whether a sounding is coupled or not. To the user of TEM measurements, the cost of accepting a coupled sounding is a lot higher than culling a non coupled sounding. The reason being that coupled soundings invalidate the later inversion, where resistivities of the earth are calculated. Culling a few extra soundings at the raw data stage, does not involve a large cost as there are a lot of measurements due to the high lateral resolution, further the data is averaged in stage 3 lowering the influence of the individual sounding.

1.2 Related Work

Currently manual inspection of data by an expert, remains the most widely used method for removal of couplings from data. Some rule based automated filters for marking capacitive couplings are also common [8], [10]. These filters rely on the gradient and sign change of a sounding. As an advantage they work independently of the specific geology. However, they lack completion and must still be manually inspected.

Reninger et al. [11] developed a user assisted method, allowing detection and filtering of couplings in SkyTEM data, based on a statistical approach. Although the instrumentation and data is different from tTEM, the problem remains the same, detecting couplings in TEM data. The method relies on singular value decomposition to find dominant principal components to reconstruct electromagnetic decays, from which noisy gates were detected. Though the method is efficient it is still user assisted, we are aiming for a completely automated process.

In an attempt to utilize neural networks Andersen et al. [6], created a simple three layer artificial neural network to detect couplings in SkyTEM data. As input for the model they used normalized gate values from three neighbouring soundings along with the altitude of the airborne TEM system to account for the change in

height above ground. For ground-based systems, the altitude is fixed and this parameter becomes superfluous for detection of coupled data. Andersen et al. achieve a hit rate of about 90%. Such a hit rate is still not sufficient and their model must be adopted to the local geological setting.

In a recent master thesis by Andreas Taankvist, 2018, in the Hydro Geophysics Group, another type of neural network for detection of couplings in tTEM data is investigated. Adopting an image based approach, he constructed a multi channel two dimensional convolutional neural network. As input images of multiple soundings and gates were used. The network was tested on both real data and synthetically generated coupled and non coupled data, achieving above 90% accuracy on the synthetic data but much lower when tested on real data.

Chapter 2

Method

This chapter describe the methodology used in the thesis work to answer the problem statement. A description of the three machine learning methods used is given, along with necessary preprocessing steps. We will start with a baseline model and from there move into more advanced models, utilizing the sequential nature of the tTEM measurements.

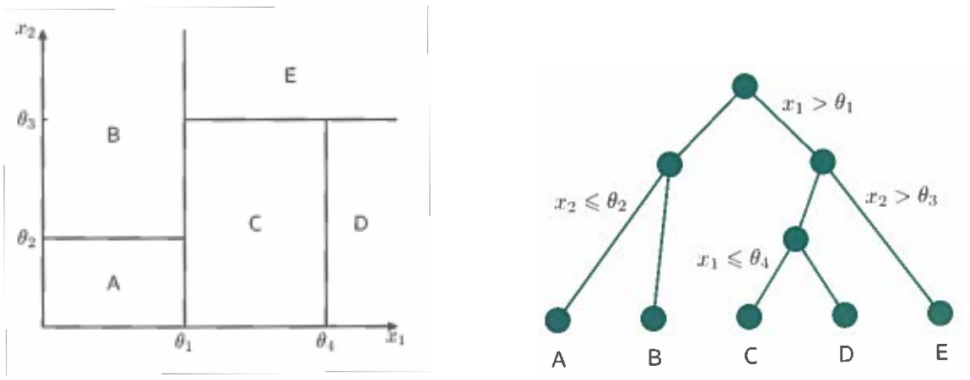
2.1 Random Forest

Random Forest is a method introduced by Kam Ho in 1995 [12] and extended by Breiman in 2001 [13], capable of performing both regression and classification, though we shall only be using it for the latter. The algorithm is widely used across many fields of machine learning and has more than 23000 citations [14]. It is known for being very effective and simple to adjust, making it a good starting point for further investigation and baseline comparison.

The algorithm is constructed from multiple binary decision tree's, hence the name 'forest'. Random Forest is an ensemble method, built on the experience that combining multiple weak classifiers often result in improved predictive performance [15]. It operates by constructing a multitude of individual decisions trees on a random subset of the feature space, a technique known as bagging. The output is the class that is the mode or mean of the individual trees. Bagging decorrelate the individual trees, by growing them on different training sets. The result is trees with a low bias but high variance. By computing the average of many trees, the random forest result in both low bias and variance [16].

A decision tree works by partitioning the input space into cuboid regions with axis aligned edges, as illustrated in figure 2.1a. Each region is assigned to a target class. The classification process of a new input sample can be described as a traversal of a binary tree as illustrated in figure 2.1b. Each node in a tree corresponds to one input feature. The root node is split according to a threshold value, dividing the input space in two. The sub regions can then be split again, until a leaf node is reached.

To learn a tree structure from a training set, we must choose an input variable at each node, as well as a threshold value for the split. Different algorithms for training a tree exist, we restrict ourselves to the Classification and Regression Tree (CART) algorithm, as it is the one used in this thesis. The algorithm recursively partition the training set in a left and right subset using a feature k and threshold t_k such that samples with same labels are grouped together. The pair (k, t_k) is chosen by minimizing the cost function C (equation 2.2). Let Q be the data at the node m, for a



(A) (B)

FIGURE 2.1: Illustration of a random decision forest. (A) partitioning of a 2D input space. (B) Binary tree corresponding to the partitioning in (A). Adapted from [17].

given pair (k, t_k), we define the partition:

$$\begin{aligned} Q_{left} &= x \mid x_k \leq t_k \\ Q_{right} &= x \mid x_k > t_k \end{aligned} \tag{2.1}$$

The cost function can then be written

$$C(k, t_k) = \frac{n_{left}}{N_m} I\left(Q_{left}\right) + \frac{n_{right}}{N_m} I\left(Q_{right}\right), \tag{2.2}$$

where N_m is the amount of data in the m'th node and $n_{left/right}$ is the number of samples in the left or right subset respectively. I is a measure of impurity. We will use the Gini impurity measure [17]

$$I(i) = 1 - \sum_{i=1}^{c} p_i^2, \tag{2.3}$$

where p_i is the fraction of samples belonging to class i in a node, c denotes the number of classes (two in our case). If all samples belong to the same class the score is 0 i.e. pure, if the classes are equally mixed the score is 0.5 i.e. impure. Ideally a leaf node should be pure.

To summarize, at each node an exhaustive search is done throughout the input space to find a feature and threshold value resulting in the lowest impurity measure for the parent node. The splitting procedure is done recursively until a stopping criteria is met, in our case when there is no longer a significant decrease in impurity. The output of a single decision tree is the probability of a sample belonging to a class, i.e. the fraction of training samples of the same class in a leaf node. The output of the random forest is the average of each tree's probability estimate.

2.2 One Dimensional Convolutional Neural Network

Based on results of the random forest model presented in chapter 3, a convolutional neural network (CNN) was chosen as a next step. Sparse connectivity of the CNN architecture, as explained below, is the key feature that make it desirable in our case.

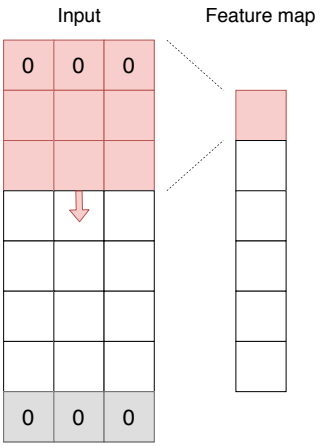


FIGURE 2.2: Schematic of one dimensional convolution. The elements containing a 0 are a result of zero-padding, to keep the feature map the same dimension as the input. The area shaded in red is the filter moving down through the input. The filter is defined by its width, in this case three and step size, in this case one.

Convolutional neural networks are essentially a type of artificial neural networks, where the layers use convolution instead of the usual fully connected weights. The convolutional layer is the basic building block of a CNN. The layer extract features from the input by convolution with a filter, resulting in a feature map as illustrated in figure 2.2. Different from a traditional fully connected layer, the convolutional layer is sparsely connected. A single element in the feature map is only connected to a small patch of input values, referred to as local receptive field. The idea being that nearby features are likely more relevant to each other, than features far away. The local receptive field is a key reason for using the CNN to solve our problem. Since tTEM data is sequential in nature, we require a context aware algorithm. Usually in a CNN architecture multiple convolutional layers are stacked. The early layers close to the input result in low level features, if it was an image it could be edges or blobs that are detected. The later layers will form higher level features based on the low level feature map, ultimately yielding information about the image as a whole. As such the convolutional layers act as a feature extraction technique. All nodes in a feature map are constrained to share the same filter, which means a feature learned in one location can also be identified in another, resulting in shift invariance of the algorithm. Additionally, it reduces the number of parameters adjusted during training, resulting in faster computation.

Formally we can write a discrete convolution of input x and filter w for a time index t as

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(a-t). \tag{2.4}$$

The filter w is also referred to as the kernel. The convolution is one dimensional in the sense that the functions are only convoluted over the time index. In practice we can implement the infinite summation as finite summation over the array elements where they are defined [18]. Furthermore, the kernel is usually flipped in the actual implementation of a convolutional layer. This is because in a CNN architecture the filters are learned, and the model would simply learn a rotated version of the filters if necessary. Thus the function actually becomes a cross correlation, which is used

interchangeably with convolution in the literature

$$s(t) = (x * w)(t) = \sum_a x(a)w(a+t). \tag{2.5}$$

A filter is defined by a kernel and stride size. The kernel size define how many input samples the filter span, the stride size how far the filter moves for the next convolution. A one dimensional convolution operation is illustrated in figure 2.2, with a kernel size of three and stride one. The image also illustrate the concept of zero-padding. Since the kernel size is three, we can pad the input with zero values to keep the feature map in the same dimension as the input.

Convolutional layers are usually followed by a standard artificial neural netowork with fully connected layers for classification purposes. The CNN designed in this thesis also utilize fully connected layers in the end for prediction. The last layer is a single neuron activated by the sigmoid activation function described in section 2.5. The output is a number between 0 and 1, interpreted as the probability of a classification belonging to class 1. A neuron is a mathematical function taking a set of inputs and sum them to produce an output known as an activation. Usually the input is weighted and the sum is passed through a non linear function known as activation function described in section 2.5.

To measure how close the output of the network, ie. the prediction, is to the target value, we make use of binary cross entropy[16], also known as log-loss. For a prediction $\hat{y}$ between 0 and 1 and a true label y either 0 or 1, we define the binary cross entropy as

$$H(y, \hat{y}) = -y \log \hat{y} - (1-y) \log(1-\hat{y}). \tag{2.6}$$

The error of a prediction increases logarithmically with the difference between the predicted value and the true target. Thus errors that are wrong and very confident are punished harder than errors that are wrong but less certain.

2.3 Autoencoder

While the first two approaches have been in the category supervised learning, the last method analyzed is based on unsupervised learning. It is motivated by an attempt to avoid issues with the manual labelling of the data sets. We will use a classifier which require only one class to make predictions. The autoencoder, also known as auto associative neural network, is essentially a form of the classic multilayer perceptron [19]. The autoencoder is illustrated in figure 2.3. The idea is to reconstruct the input data at the output of the network. The target variable is therefore the input samples, eliminating the need for labelled data making the method unsupervised. The input will only consist of the non coupled class, thus it should learn how to reconstruct non coupled samples, but not samples showing coupling.

The autoencoder project the input onto a latent space by learning an encoding of the data. The encoding is followed by a decoding, where the autoencoder learn a reconstruction from the data representation in the latent space, as close as possible to the original input. In the case where the latent space is of a lower dimension than the input space, as is illustrated in figure 2.3, the autoencoder is called under complete. The undercomplete autoencoder is the type we will work with in the thesis. Since the latent space is of lower dimensionality, the autoencoder essentially perform dimensionality reduction, which is one of its use cases [19].

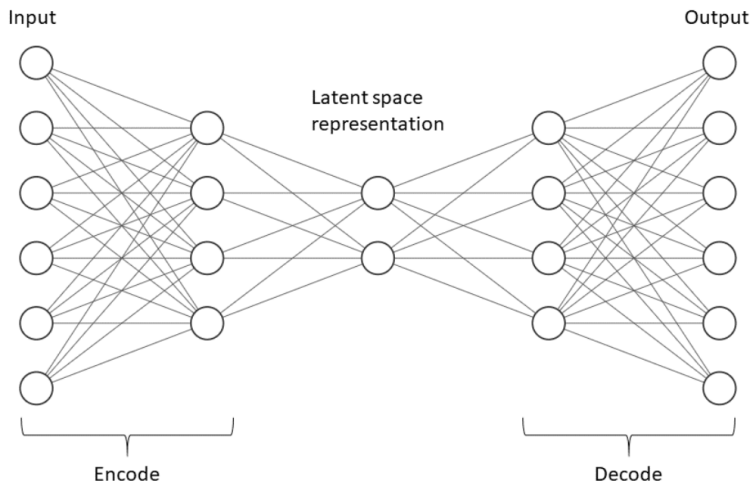


FIGURE 2.3: Illustration of the autoencoder architecture.

Formally the autoencoder takes an input $x \in \mathbb{R}^d$ and maps it to the latent representation $h \in \mathbb{R}^{d'}$, using a deterministic function

$$\mathbf{h} = f_\theta(x) = \sigma\left(\mathbf{W}x + \mathbf{b}\right) \tag{2.7}$$

with parameters $\theta = \{\mathbf{W}, \mathbf{b}\}$. Where W is a $d' \times d$ dimensional weight matrix, and b is a d' dimensional offset vector. σ denotes the activation function explained in section 2.5. Equation 2.7 is the coding step, used to construct a code from the input. To reconstruct the code, a reverse mapping of f is computed

$$\mathbf{y} = f'_\theta(h) = \sigma\left(\mathbf{W}'h + \mathbf{b}'\right) \tag{2.8}$$

with $\theta = \{\mathbf{W}', \mathbf{b}'\}$. The parameters are optimized during training, using back propagation, by minimizing an appropriate cost function [20]. Specifically we shall choose the mean absolute error

$$MAE = \frac{1}{2}\sum_{i=1}^{N}|y_i - x_i|. \tag{2.9}$$

When compared to mean squared error, MAE is less sensitive to outliers and there is no reason to give extra weight to extremities in our case.

2.4 Training and Optimization

The CNN and autoencoder are both neural networks and follow a similar training and optimization strategy. In both cases we use back propagation following a gradient decent approach to train the networks. We will give a short description of the algorithm, as the finer details of the the two methods are already well described in the literature and the purpose of the thesis is not to investigate the underlying algorithms commonly used in training neural networks.

Backpropagation is an algorithm used to update the weights of a network. In the context of neural networks, what is meant by 'learning' is to find weights and biases that minimize a certain loss function. Initially the weights of a network are usually chosen at random. Backpropagation can be described in four steps:

1. A forward pass where the input is passed trough the network and the output of the network is computed.

2. An error between the network output and the target value is calculated. This is done by using a loss function. In our case the binary cross entropy is used for the CNN and mean absolute error for the autoencoder as described.

3. The backward pass iterate backwards through the network, to calculate how large influence the weights in each layer have on the loss of the output and how they should be changed to minimize it.

4. In the final step, the weights are adjusted proportional to the negative gradient of the loss function, ie. a gradient descent.

The forward and backward pass is done for all input samples after which the weights are updated. A full backpropagation computed on all input samples is known as an epoch. For computational efficiency, the input is often randomly shuffled and divided in smaller batches. The weights of the network are then updated after backpropagation on each batch. One epoch is then when all batches have contributed to updating the weights and biases. Several different approaches to gradient descent exists, we shall make use of the popular and relatively new Adam algorithm [21].

We distinguish between parameters learned by the model during backpropagation and parameters externally adjusted, usually by the user, known as hyperparamters.

2.5 Activation Functions

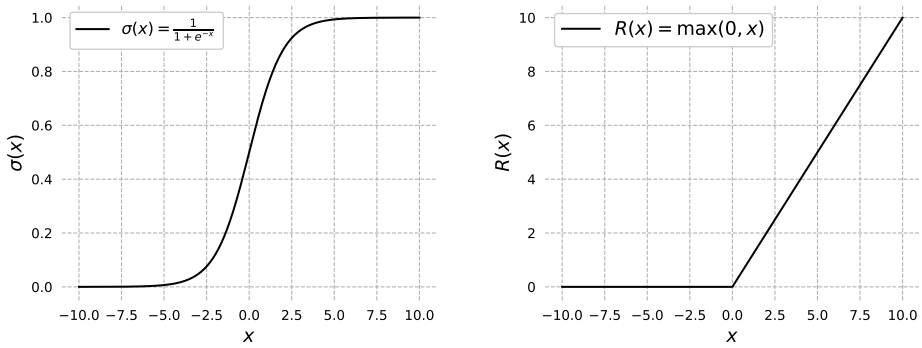
Both the CNN and autoencoder are neural networks where the neurons are activated by an activation function. A neuron calculate the weighted sum of its input plus a bias. An activation function is applied to the neuron to decide whether it should 'fire' or not i.e. if the value is passed along to the next layer and to what extent. We will make use of two common activation functions; sigmoid and rectified linear unit (ReLU). The sigmoid function is illustrated in figure 2.4a and defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.10}$$

The output is a number between 0 and 1, particularly useful for binary classification, as is the problem of the thesis. The sigmoid function is a cumulative distribution function of a logistic distribution and is thus interpreted as a probability. The ReLU function, illustrated in figure 2.4b, is defined as

$$R(x) = \max(0, x). \tag{2.11}$$

The ReLU avoid the vanishing gradient problem of the sigmoid. As x becomes very large the gradient of the sigmoid vanish, which is problematic for the backpropagation step. The gradient for ReLU is constant for large x. For negative x the output is zero, resulting in fewer active neurons, improving training speed. At $x = 0$ the ReLU is not differentiable, but in machine learning implementations it is usually defined to zero.



(A) Sigmoid function. (B) ReLU function.

FIGURE 2.4: Illustration of two activation functions.

2.6 Preprocessing

In any machine learning task, preprocessing of the data is a crucial step [22]. We will make use of two techniques as described below, data normalization and data sampling which may be regarded as a type of feature engineering.

Selection: The tTEM data used in the process of this thesis contains a total of 33 gates in the high moment. We will only be using gate 8 trough 24. The first seven gates are never used because they constitute the very early times after turn off of the primary field and are therefore biased by the transmitter coil. The last few gates are dominated by noise and contain no useful signal. Research to improve signal to noise ratio in the late gates is ongoing. The low moment contain only 4 useful gates and correspond to the first 10 meters of the surface, for simplicity we do not use the low moment.

Data normalization: The magnitude of the measured dB/dt values decrease orders of magnitude with increasing gate number. To most machine learning algorithms this pose a problem, as the later gate times will be insignificant during the gradient descent compared to the much larger values. We will bring all gate values to the same scale by computing the z-score, a common normalization strategy in machine learning. The z-score is defined by

$$z = \frac{x - \mu}{\sigma}. \tag{2.12}$$

where x is a sample-, μ is the mean- and σ the standard deviation of a gate. Effectively we are shifting the mean of each gate distribution to zero and scale the dB/dt values such that the standard deviation of the distribution is one.

Sampling: To achieve a higher degree of context in the input space, we will make use of a windowing strategy. The data sequence is sampled using a moving window, selecting 20 soundings at a time with a stride size of 3, resulting in an overlap of 17 soundings per window. The dimension of the sampled data segment is then 20 soundings by 17 gates. The size and stride size of the window has been optimized through a manual search. These matrices, referred to as windows, will be the input to the one dimensional convolutional network and the autoencoder. In the case of the CNN, a label is needed for supervised learning. The annotation of a window is chosen as the label value of the sounding in the middle of the window. As the data windows have an overlap of 17 soundings, each sounding will receive multiple

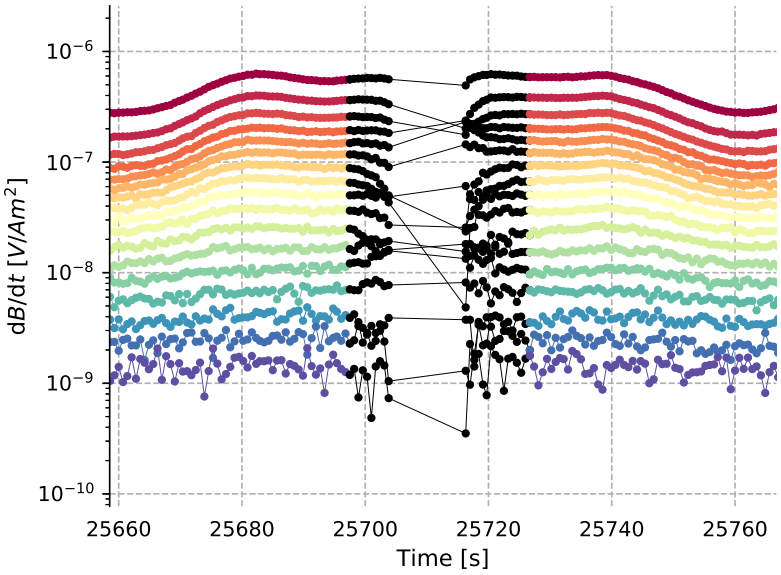


FIGURE 2.5: Example of a hole in data recordings due to the tTEM system turning a corner. Each color represents a gate, lower gates have longer gate times. Black represent manually labelled couplings. On the edges of the hole, some soundings have been marked, even though they show no direct sign of capacitive or galvanic coupling.

predictions as they are part of multiple windows. The final prediction of a sounding is the average of all predictions of that sounding.

Furthermore, we remove some data from the original tTEM data set, before any models are trained on the data. When the tTEM system turns a corner, the recorder is turned off. The turning cause a corruption of data, because the geometry of the setup is changed [3]. The turn-off results in 'holes' in the data, where no measurements are recorded. An example is given in figure 2.5. The edges around these holes are problematic as the turn on and off is not consistent. Some soundings around the holes therefore show an unnatural signal, hence they have been labelled as coupled in the manual processing in order to not use them in the further inversion. However, these soundings are not actual couplings, but just an artifact of the system turning, thus to limit this influence 20 soundings to the left and right of all holes are removed.

2.7 Performance Evaluation

A characteristic of the tTEM data used, is that the proportion of coupled to non coupled data varies significantly across different survey areas, leading to class imbalance. In the common case we would expect the non coupled class to be dominant, since coupled data is not useful and areas that are known to be dominated by man made conductors would be avoided. Changes in the class distribution from one area to another is not unrealistic, as the location of underground conductors such as wires and piping is not generally known. Thus we have to account for this imbalance when evaluating the performance of a predictor.

The overall accuracy will be presented, but gives limited insight due to the class imbalance. In most dataset the number of non coupled soundings outweigh the

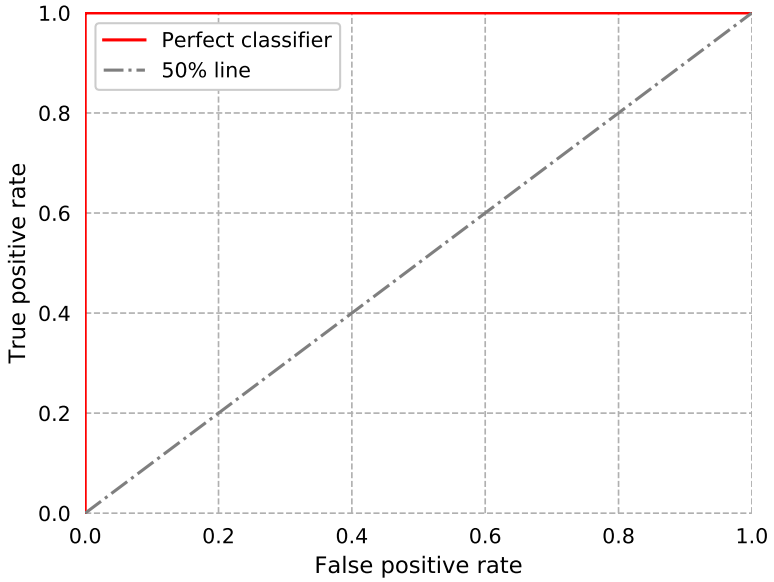


FIGURE 2.6: Illustration of a ROC curve.

number of coupled soundings between 7 in 10 to 9 in 10. More informative measures for imbalanced data is the precision and recall measures and the receiver operating characteristic (ROC) curve [23]. In this section we will regard the coupled samples as the negative class and the non coupled samples as the positive class. Let TP denote true positives and TN true negatives, if P and N is the total amount of samples in the positive and negative class respectively, we can define the accuracy as

$$Accuracy = \frac{TP + TN}{P + N}. \tag{2.13}$$

Naturally if a data set includes only 10% of the minority class, and 90% of the majority, a naive classifier will simply classify all samples to the majority class, achieving an accuracy of 90%. At face value 90% accuracy seems successful, but it fails to reflect the fact that zero of the minority class samples were correctly classified. The inherent problem is that the metric is sensitive to the class distribution, since it depends on both classes [23]. To counter this effect, another frequently used metric is precision and recall

$$Precision = \frac{TP}{TP + FP} \tag{2.14}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{P}. \tag{2.15}$$

Intuitively precision is a measure of exactness, i.e. of the samples classified as positive, how many were true positives. Recall is a measure of completeness i.e. how many samples of a particular class is identified. It is easily seen that while precision, like accuracy, is dependent on both classes and thus distribution dependent, recall is not.

The Receiver operating characteristic visualize the trade off between true positive rate (TPR), which is defined in the same way as recall, and false positive rate (FPR). An example is found in figure 2.6. TPR and FPR can be written as

$$TPR = Recall = \frac{TP}{TP + FN} = \frac{TP}{P} \tag{2.16}$$

$$FPR = \frac{FP}{TN + FP} = \frac{FP}{N}. \tag{2.17}$$

It can be used for binary classifiers with a probabilistic or ranked output, where a threshold is used to make the output binary. For all possible threshold values, we plot recall against FPR. A perfect classifier will end up in the upper left corner, with a TPR of 1 and FPR of 0. A random prediction will end up on the diagonal line, known as line of no discrimination, regardless of the distribution of the positive and negative class. For example, if we randomly predict the positive class 80% of the time, the classifiers ROC point will tend to $(0.8, 0.8)$, for an increasing sample size. Points below the line of no discrimination, is a predictor worse than random guessing, however the predictor can simply be negated to obtain good results. An attractive property of ROC curves is their insensitivity to changes in class distribution. The TPR and FPR is a ratio between each of the two classes, thus if the proportion of positive- or negative samples change in a test set, the ROC curves will not [24].

Finally, the area under the ROC curve, denoted AUC in the following, is a measure of a predictors ability to discriminate the two classes. Since AUC is a portion of the area of a unit sqaure, the values will be between 0 and 1, with 1 being a perfect classifier. However, because random guessing produce the diagonal line and thus an AUC of 0.5, lower AUC scores are rarely reported. AUC is interpreted as the probability that a randomly selected positive sample will be ranked higher than a randomly selected negative sample.

2.8 Cross validation

Cross validation is a statistical method commonly used in machine learning to estimate the predictive capabilities of a model. In particular we shall make use of k-fold cross validation. The original data set is split into a training and a test set. The test set is reserved for final evaluation. The training set is then split into k sub samples, of which one is retained for validation and the remaining $k - 1$ is used for training as illustrated in figure 2.7. The model is trained and evaluated for k iterations, such that each of the k subsets are used exactly once as a test set. We then report the mean of the evaluation metrics across the k evaluations. We use cross validation to tune the hyper parameters of the model and to detect whether the model is over- or underfitting. Because the network is not directly trained on the validation set, it provides an unbiased view of the performance. However, the validation set indirectly affect the final model since it is used to adjust the hyper parameters of the model. To assess how the model generalizes to unseen data, a final evaluation is done on the test set. This also makes it possible to compare different algorithms as none of them have been tuned particularly to the test set [**Pedregosa20193.1.Performance**].

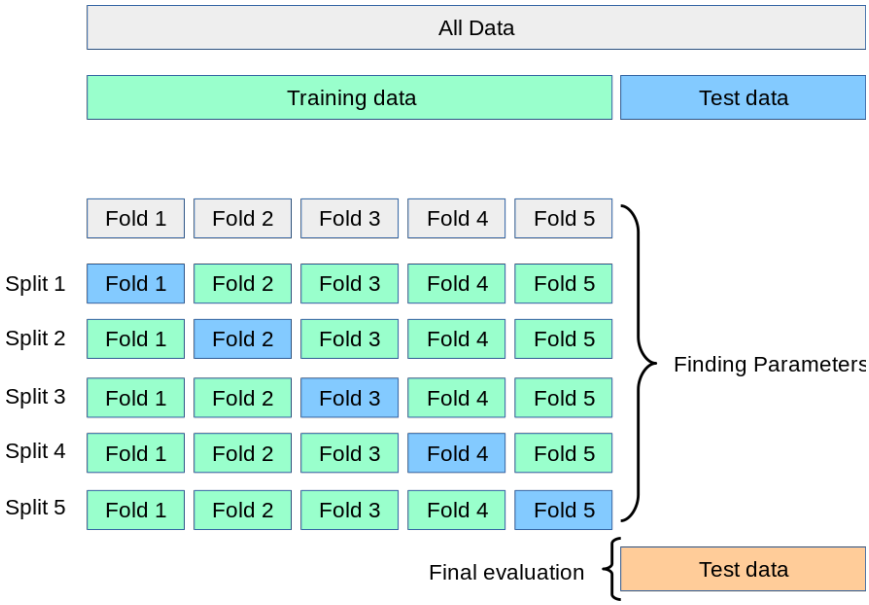


FIGURE 2.7: Illustration of a five fold cross validation. Image adapted from **[Pedregosa20193.1.Performance]**.

Chapter 3

Results and Analysis

This chapter present and analyze the results of the three previously described models. The results presented constitute precision-, recall- and AUC score from a five fold cross validation, calculated as the mean and standard deviation of the five iterations. Furthermore the ROC curves and a histogram of the score distribution from each predictor on each cross validation iteration is presented. Finally in Section 3.6 the accuracy-, precision-, recall- and AUC score of the final evaluation of the three models against the test set is presented for comparison, along with the ROC curve of each model evaluated on the test set.

3.1 Programming Environment

The models described in chapter 2 were developed in the Python 3 programming language. Specifically, the random forest algorithm were implemented using the scikit-learn API [**PedregosaF.andVaroquauxG.andGramfortA.andMichel2011Scikit-learn:Python**], a machine learning library for the Python programming language. The 1D-CNN and the autoencoder were implemented using Keras [25], a high level neural networks API, running on the Theano backend. Theano is a Python library and optimizing compiler for manipulating and evaluating mathematical expressions [26]. The code base is documented and available on the GitHub repository [27]. Computations were primarily carried out on a standard desktop intel core i5 CPU. Computation times for a single training and test iteration ranged from about to 1 minute for the random forest, slightly longer for the autoencoder and 2-3 minutes for the 1D CNN.

3.2 Data sets

The data used for training and testing the machine learning models have been processed according to section 2.6. The entire data set is split into 80% training set and 20% test set reserved for final evaluation. To manually adjust hyper parameters during training, a five fold cross validation is performed on the training set as described in section 2.8. The training set is split 5 times in two subsets, 20% validation and 80% training data. The data is always normalized using the statistics of the training set to calculate the z-score (equation 2.12) for both the training and test set. If the statistics are calculated over the entire data set, information from the test set would leak into the training set.

The data is from a survey done in Stendalmark, Denmark between 20. to 21. November 2018. The set consist of 33064 soundings, each with 17 gates. Coupled soundings constitute 24% or 7936 samples distributed on 116 coupled segments. After splitting, the training set consist of 26451 soundings. Coupled samples constittue

22% or 5875 samples distributed on 86 segments. The test set consist of 6613 soundings. Coupled samples constitute 31% or 2061 samples distributed on 30 segments.

3.3 Random Forest

3.3.1 Implementation

The random forest as described in section 2.1 is constructed with 100 trees and grown from the training data based on the Gini criterion (Equation 2.3). The input to the model will consist of a single sounding, that is, a vector of 17 dB/dt values and the corresponding label. The model output a score between 0 and 1 for each sample, interpreted as the probability of a sample belonging to class 1 i.e. non-coupled. The probability score is converted to a binary score, by choosing a threshold above which a sample is classified as non-coupled and below as coupled. The threshold is chosen to 0.7 based on the score distribution, as explained in the following section. The most important hyper parameter of the random forest model, is the number of trees and how many features to consider before making a split. In this case the performance is converged after 100 decision trees and the full feature space is searched through ie. the 17 gates when looking for the best split. The parameters have been manually determined based on cross validation results. A characteristic of random forest is that a default set of hyper parameters usually works well [28], thus remaining parameters are kept at these values as implemented in [**PedregosaF.andVaroquauxG.andGramfortA.andMichel2011Scikit-learn:Python**].

3.3.2 Cross validation results

Precision and recall scores of the random forest algorithm from a five fold cross validation on the training set is presented in table 3.1.

TABLE 3.1: Random forest mean and standard deviation ( ) of k-fold cross validation.

	Precision [%]	Recall [%]	Samples
Coupled	63 (21)	81 (3.1)	1175 (752)
Non coupled	94 (4.1)	89 (9.4)	4115 (752)

As early results the random forest is promising, especially the non coupled class is doing well. However, a higher, recall especially of the coupled class, is preferred since the cost of including a coupled sounding outweighs the cost of culling a non coupled sounding as described in section 1.1.1. More problematic are the high standard deviations. Clearly, performance of the five different evaluations of the network vary greatly. From the standard deviation of the number of samples in the two classes, it is clear that there is a large difference in the number of coupled samples in the test set between the five iterations. The reason is primarily the splitting technique of the k-fold cross validation. The coupled samples are not distributed evenly across the data set, thus when sub sampling the training set in five validation sets, there is a high selection bias. How well the training set represent the validation set is strongly dependent on the number of coupled samples in the two, for example if there are no or few couplings in the part of the data chosen for training, the network will not be able to learn the general structure of coupled data. The greatest standard deviation is from the precision of the coupled class. Precision is dependent

on the class distribution, thus in the case where there is a big distribution difference between the validations, the deviation of precision will necessarily be a lot higher.

In figure 3.1 a bar plot show the fraction of non coupled samples in the validation- and training set for each iteration. Evidently, in some iterations the difference between the number of non coupled samples vary a lot, as does the distribution between the iterations. Especially iteration 1 and 3 are problematic because the validation set is either completely dominated by coupled samples or non coupled. Iteration 2, 4 and 5 are however closer to the natural distribution of the entire data set with 24% coupled samples.

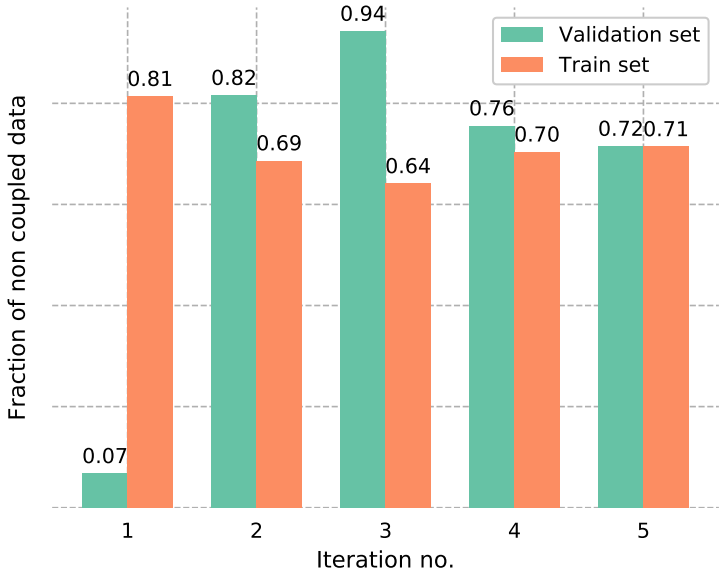


FIGURE 3.1: Fraction of non coupled samples in the test- and training set of each cross validation iteration.

Looking at the ROC curves of each cross validation iteration in figure 3.2, the variance is less explicit. The mean AUC score of the five iterations is $92(4.4)$. ROC curves are insensitive to changes in class distribution in the validation set, which means the curves will not change as a result of changing imbalance in the validation set. As observed, the curves are quite similar with the exception of iteration 5. Since the ROC curves do not change with changing distribution in the test set, the lower AUC score must be due to a lack of the training sets representational power of the validation set in iteration 5. This could be due to a different geology present in the validation and training set.

The data treated in the thesis have already been manually processed, therefore a resistivity map of the area is available. In figure 3.3, resistivities in the survey area are shown at four different depths. From the images it is clear that the resistivities and therefore the geology in the area vary a lot, even within short distances. White polygons approximately show the division of the data set into a test set, and the five polygons constituting the training set, each number indicate the area of the respective validation sets. At each depth, the resistivities is quite different within each of the data splits. Along with the differing distribution in the number of coupled samples in the training and test set, the changing resistivities can explain the quite large standard deviations in the cross validation.

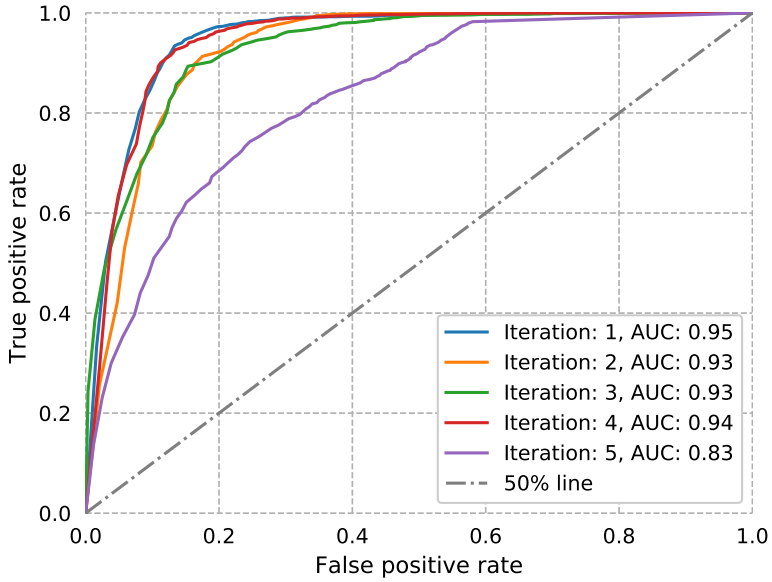


FIGURE 3.2: ROC curves of the random forest model.

TABLE 3.2: Results of random forest with random sample cross validation.

	Precision	Recall	Samples
Coupled	77 (0.9)	91 (0.8)	1175 (27)
Non coupled	97 (0.1)	92 (0.5)	4115 (28)

To further the argument that the splitting suffers from selection bias, a random sampling technique is used. To achieve more balance in the training and validation set with respect to geology and couplings, the full training set is randomly shuffled before sub sampling it in training- and validation sets. In table 3.2 results of the same random forest is shown, but this time evaluated using cross validation on the randomly shuffled data set. Clearly the performance is superior achieving recall scores above 90% and a mean AUC of 97 (0.2). The standard deviations are significantly lower than previously, suggesting that selection of training and validation sets, when sequentially sampled, are strongly biased, primarily due to the very different geology in the survey area and how the coupled and non coupled soundings are distributed.

Unfortunately a random sampling strategy is not viable due to the sequential nature of the data. It is not reasonable to include random soundings from a test area in the training set. In a production scenario the algorithm would be trained on data from one or more areas and then retrieve predictions on a new unseen area. One could however argue, that sequentially segmenting a very large data set would be equivalent to randomly segmenting a small data set, since a very large data set would potentially represent a broad geology and thus the type of geology represented in the training- and validation set would not depend on which segment it is selected from, equivalent to a randomly sampled set. Obtaining a very large data set however, raises a so far unadressed problem; generalization. Geology of the subsurface and thus the shape of soundings vary widely in space, even across a field

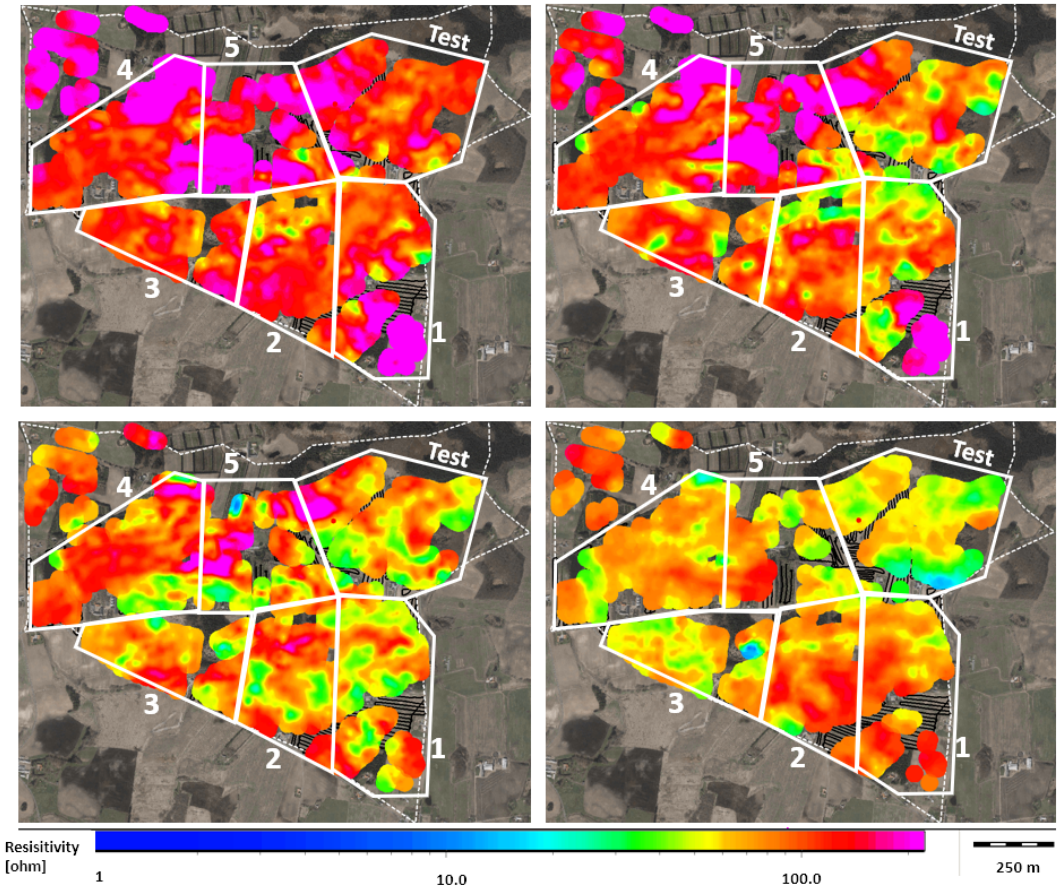


FIGURE 3.3: Map of resistivities in the survey area at different depths. Top left: 5 to 10 m. Top right: 15 to 20 m. Bottom left: 25 to 30 m. Bottom right: 50 to 60 m.

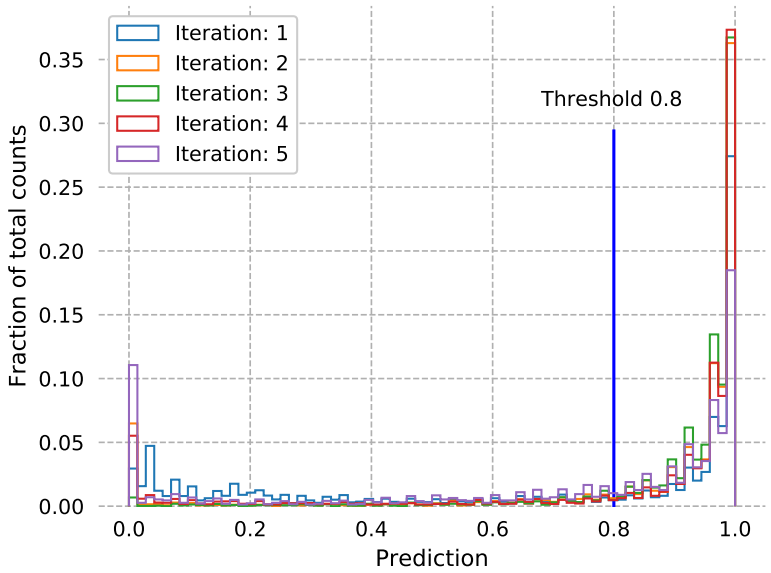


FIGURE 3.4: Histogram of scores predicted by the random forest model. A total of five histograms is shown on top of each other, one for each iteration of cross-validation.

as could be seen in figure 3.3. Training a network on data from one location and evaluating it on another might prove difficult. Andersen et al. [6] draw the same conclusion in their work. In the next sections we will investigate whether a machine learning algorithm can benefit from including context information in the algorithm. If the data is permuted at random, the sequential information is lost, thus we shall continue using sequential cross validation segmentation and because my previous argument for random sampling approximating a large sequential samplingn has not been tested.

Naturally the precision and recall depend on the threshold value of the predictions. In figure 3.4 a histogram of the predictions of the random forest for all five iterations is found. From the histogram a threshold value is chosen indicated with a blue vertical line. Ideally the predictions should only be distributed in the zero and one bins. In reality they are not and we can see the effect of increasing or decreasing the threshold. A high value would result in more false negatives and thus a higher recall but lower precision of the coupled class. Usually this would also mean a lower recall of the non coupled class. A threshold favouring a higher recall of the coupled class over the non coupled is decided, as argued in section 1.1.1.

To investigate whether our model could benefit from more data, a learning curve is presented in figure 3.5. 50 random forest models are trained, using from 1% to 100% of the training set. The validation set is the last fifth of the training set, equivalent to iteration 5 of the cross validation. The size of the training set is increased sequentially from the beginning to the end. For the first iteration using up to 20% of the training data, the curve is very odd. The oddity is attributed to the training size being very small in the beginning, in fact no larger than the validation set. Making statements about the skill of the classifier with so little data is meaningless. After about 20% the model is clearly learning. At the full training set size, the learning curve is still not converged, suggesting that more data would improve learning.

While the performance of the random forest is not bad, it is not near a required

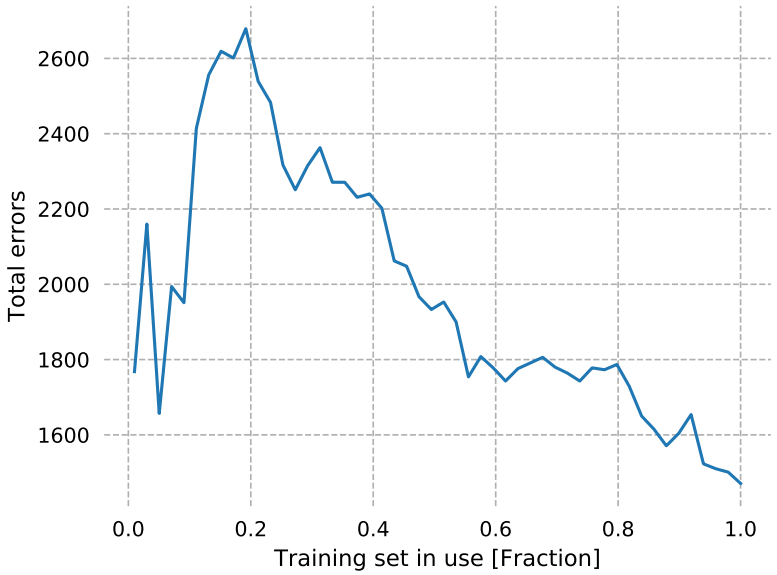


FIGURE 3.5: Learning curve of the random forest, showing the decrease in classification error as a function of increasing training set size.

operating accuracy. However, some interesting points can be learned for further development. In figure 3.6 an example of a classified segment is shown, representative of the general case. When looking at which samples are misclassified, two trends are observed. First, soundings that show very clear coupling, e.g. sign changes are usually found. Second, many single samples are misclassified, while their neighbourhood region is correctly classified. Due to the high spatial resolution of the tTEM system, we never expect a single sounding to be coupled without the neighbouring soundings also indicating a coupling. In other words, the context of a sounding is very important for accurate detection of couplings as is also suggested by Auken et al. [8] and Reninger et al. [11]. While our random forest use only one sample as input, Andersen et al. [6] attempt to increase the context awareness by representing a sounding by itself and its two neighbours. Increasing the number of neighbouring soundings included in the input does result in more information available to the algorithm, although it also increase the dimensionality of the feature space, leading to slower computations and possibly an overflow of superfluous information. Based on these considerations two strategies will be attempted:

(a) To achieve a higher degree of context in the input space, the sampling technique described in 2.6 will be used.

(b) Make use of an algorithm which is context aware by design and is able to ignore superfluous information in the increased feature space.

In the next section a convolutional neural network is investigated as it achieves goal b) when combined with the sampling technique of goal a).

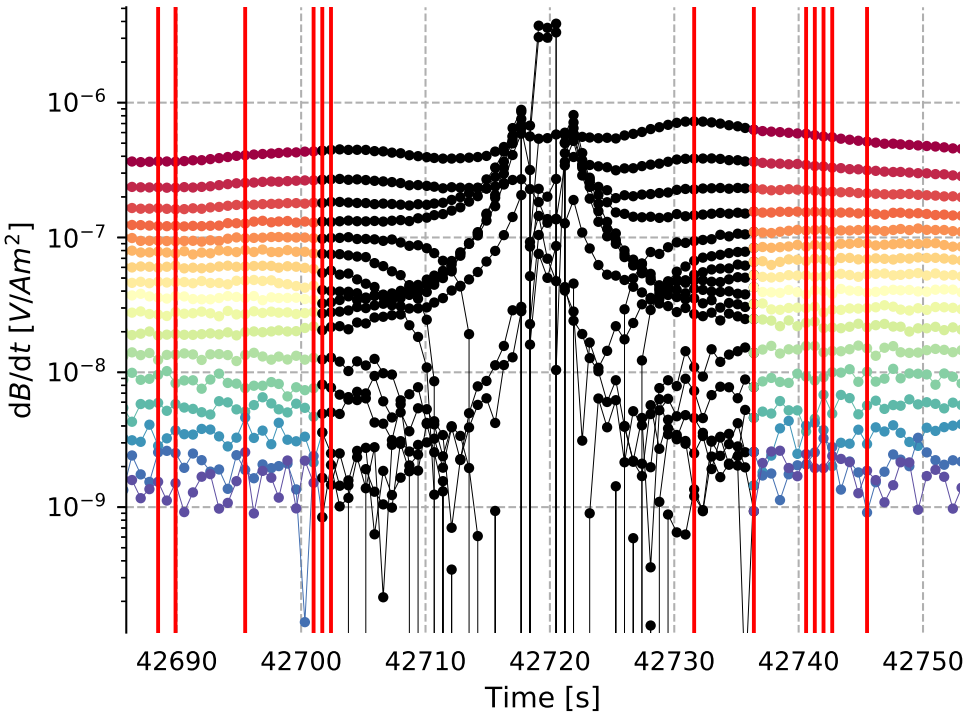


FIGURE 3.6: A segment of data classified by the random forest. Each colored line is a gate. Black dots are gate values manually marked as coupled. Vertical red lines are soundings incorrectly classified by the random forest, all soundings without vertical red lines have been correctly classified. The main part of the coupling is identified. The errors are primarily in the transition region and from single soundings being misclassified opposite to their neighbourhood. The black lines entering the bottom of the image is a result of negative dB/dt values, since it is a log-y plot.

3.4 One Dimensional Convolutional Neural Network

3.4.1 Implementation

A simple one dimensional convolutional neural network consisting of two convolutional layers is constructed, followed by three fully connected layers. We use convolutions in one dimension, because we are only interested in detecting a coupling from one sounding to the next and not the specific gate value a coupling occur. That is, the filter span all gate values and 3 to 6 soundings, and move in the direction of new soundings. A schematic of the model is shown in figure 3.7. Between the last convolutional layer and the first fully connected layer is a flattening layer, which reshape the two dimensional output of the convolutional layer to one dimension. To prevent overfitting, two dropout layers are inserted between the fully connected layers. The dropout layer simply randomly ignore 50 % of the output neurons of the previous layer during a particular forward or backwards pass. This force the next layer to learn from a varying input preventing interdependent learning between the neurons.

The convolution layers are defined from the number of filters, their size and the step size. A convolutional layer with zero-padding thus has a dimension equal to the number of input samples times the number of filters. The fully connected layers, also known as dense layers, are only defined by the number of neurons. The parameters of the CNN are summarized in table 3.3. The network was trained for 64 epochs with a batch size of 1000. Hyper parameters are chosen from a manual search based on the cross validation. The weights of the network, including the filters of the CNN layers, are iteratively adjusted using the Adam algorithm [21], an extension to gradient descent, by method of back-propagation as described in section 2.4.

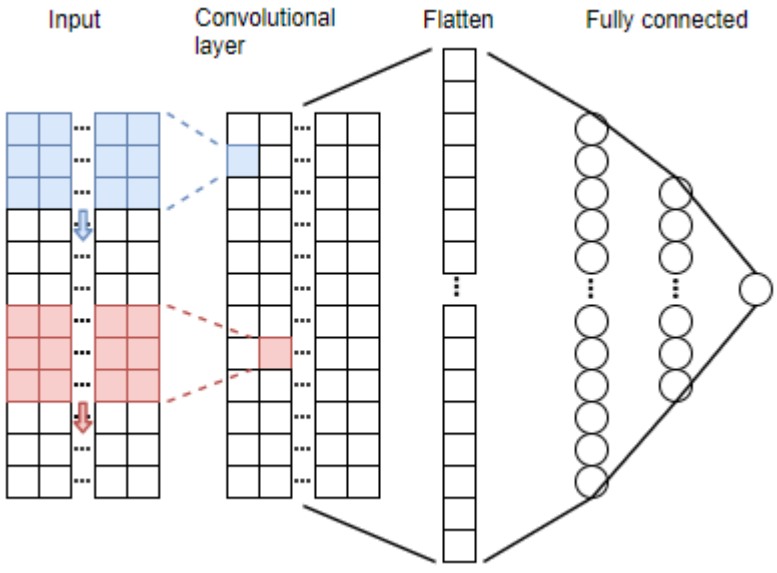


FIGURE 3.7: Schematic of a 1D CNN. The input layer is convoluted with different filters, represented by the blue and red colors. The filters have a kernel size of three and stride size of one. Each filter convolution result in a feature vector. At the end are fully connected layers computing the output of the network.

The network is trained on the data presented in section 3.2. The input is the data window obtained by using the sampling technique described in section 2.6 and thus

each input has dimension 20 soundings by 17 gates. The output of the network is a number between 0 and 1 computed from the sigmoid activation function, describing the probability of a window belonging to the the non coupled class. The loss of a prediction is measured using binary cross entropy. As with the random forest model, the output is converted to a binary by choosing a threshold value of 0.87, based on the distribution of predictions in figure 3.9 and a balance between the recall of the coupled- and non coupled class.

TABLE 3.3: 1D CNN layout.

Layer	Filters/Neurons	Kernel size	Step size	Padding	Activation
1D-CNN w. padding	100	6	1	yes	ReLU
1D-CNN w. padding	100	3	1	yes	ReLU
Fully connected	100	-	-	-	ReLU
Droput 50%	-	-	-	-	-
Fully connected	50	-	-	-	ReLU
Droput 50%	-	-	-	-	-
Fully connected	1	-	-	-	Sigmoid

3.4.2 Cross validation results

Precision and recall from the cross validation of the 1D-CNN is presented in table 3.4. Results of the CNN show some improvement over the random forest. The standard

TABLE 3.4: 1D CNN results and standard deviation ( ) from cross validation.

	Precision	Recall	Samples
Coupled	69 (14)	89 (4.8)	1175 (752)
Non coupled	96 (2.9)	91 (2.5)	4115 (752)

deviation on the number of samples in the coupled and non coupled classes remain the same, as the same splits for cross validation are performed, which also imply the same problem of selection bias in the training and test set. The ROC curves in figure 3.8 show a similar pattern as the ROC curves of the random forest. The AUC however, is improved with a mean of 95 (2.7), directly indicating that the predictors ability to distinguish a randomly selected coupled sample from a randomly selected non coupled sample is improved at a probability of 95%.

In figure 3.9 a histogram of the predictions of the CNN is shown along with the selected threshold. The histogram also indicate a confident separation of the two classes with a majority of the samples predicted close to either 0 or 1 and relatively few receiving a prediction in between.

Figure 3.10 shows the performance of the network in iteration 5 of the cross validation as measured by the loss of the model per training iteration, for both the training and validation set. The figure is representative of all five iterations. The network is converged after the employed 64 epochs, but a gap remains between the validation and training set. This can be interpreted as the training set not sufficiently representing the validation set, resulting in a higher loss on the validation set.

Looking at which samples are misclassified, it is consistently found to be soundings in the transition between non coupled and coupled segments, as was also the

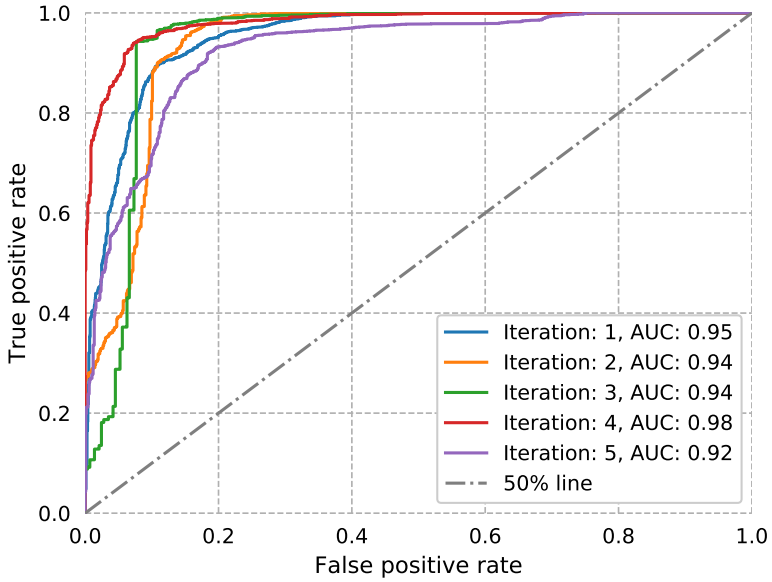


FIGURE 3.8: ROC curves for the CNN.

case of the random forest model. An example is given in figure 3.11, where a segment of data containing a single coupling is shown. It can be seen that the major part of the coupling itself is identified, but the transition from non coupled to coupled data does not match the target. It is not surprising since the transition from non-coupled to coupled measurements is gradual, making it impossible to definitively state exactly where the transition occur. Furthermore, the annotation of coupled data is highly biased according to the experience and production concerns of the operator.

The windowed strategy did improve on the problem with single soundings being predicted as opposite of their neighbourhood and the CNN also improved the performance metrics though not satisfactory. We will therefore continue with the windowed data sampling, but another type of algorithm in an attempt to address the problem with lack of representational power in the training data and the annotation problem. We change our focus from supervised learning to unsupervised. To factor out the human error in the annotation of the training data, we will use a model that is trained purely on non-coupled data. Furthermore, this eliminate any dependency on the distribution of classes in the training input, as the algorithm only need to learn one class.

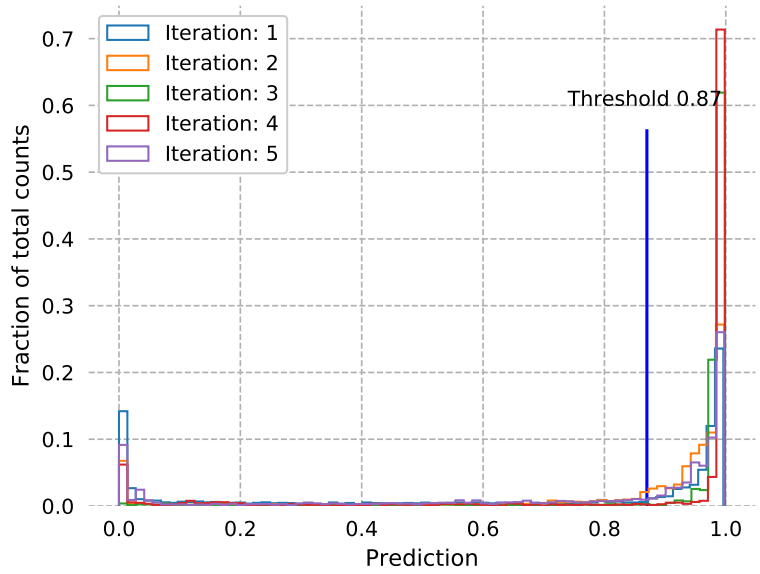


FIGURE 3.9: Score distribution for the CNN. Vertical blue line indicate the chosen threshold.

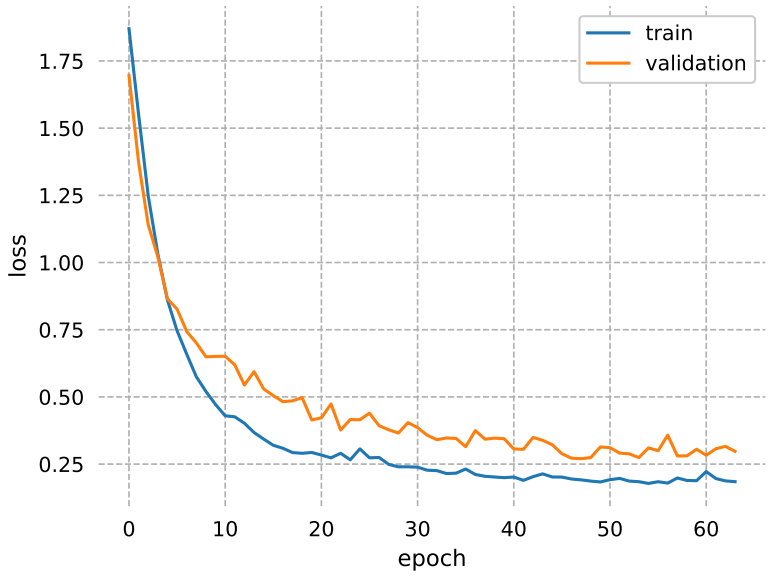


FIGURE 3.10: Loss curve of the CNN for iteration 5.

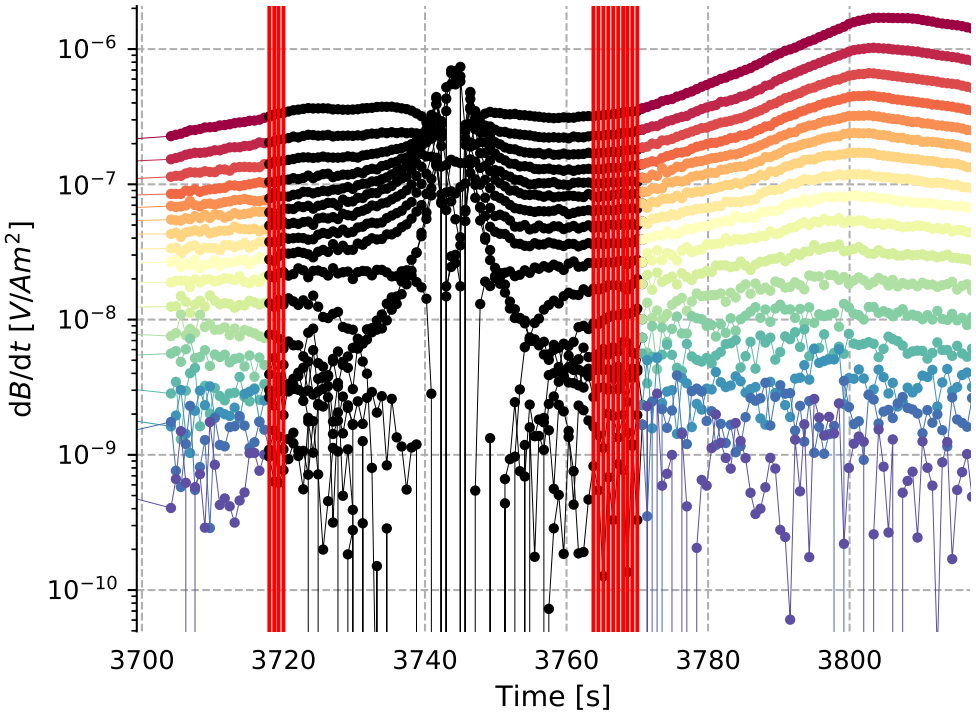


FIGURE 3.11: A segment of data classified by the 1D CNN. Each colored line is a gate. Black dots are gate values manually marked as coupled. Vertical red lines are soundings incorrectly classified by the CNN, all soundings without vertical red lines have been correctly classified.

3.5 Autoencoder

3.5.1 implementation

The autoencoder is constructed from five fully connected layers. The number of neurons in the first and last layer are equal to the dimension of the input samples. The hidden layers are symmetrical and constructed in an under complete manner, with fewest nodes in the middle layer. The network topology is summarized in table 3.5. Since we are constructing an under complete autoencoder, the latent space must necessarily be of lower dimension than the input space. The network is trained for 100 epochs with a batch size of 64. Parameters were adjusted manually based on the cross validation results. The weights of the network are iteratively adjusted using the Adam algorithm [21], by method of back-propagation.

As with the other models, the network is trained on the data described in section 3.2. The data is sampled in data windows as described in section 2.6, of size 20 soundings by 17 gates. The window is reshaped into a vector by stacking each sounding on top of each other, thus the input is a vector with $17 \cdot 20 = 340$ entries. The target is the same at the input sample. The output of the network is a dissimilarity measure between the network output and the target sample, as defined by the mean absolute error (equation 2.9). A binary decision is made by choosing a minimum acceptable error for non coupled data. The threshold is chosen based on the distribution of prediction errors figure 3.14, such that samples with a MAE below 0.1 were classified as non coupled.

3.5.2 Cross validation results

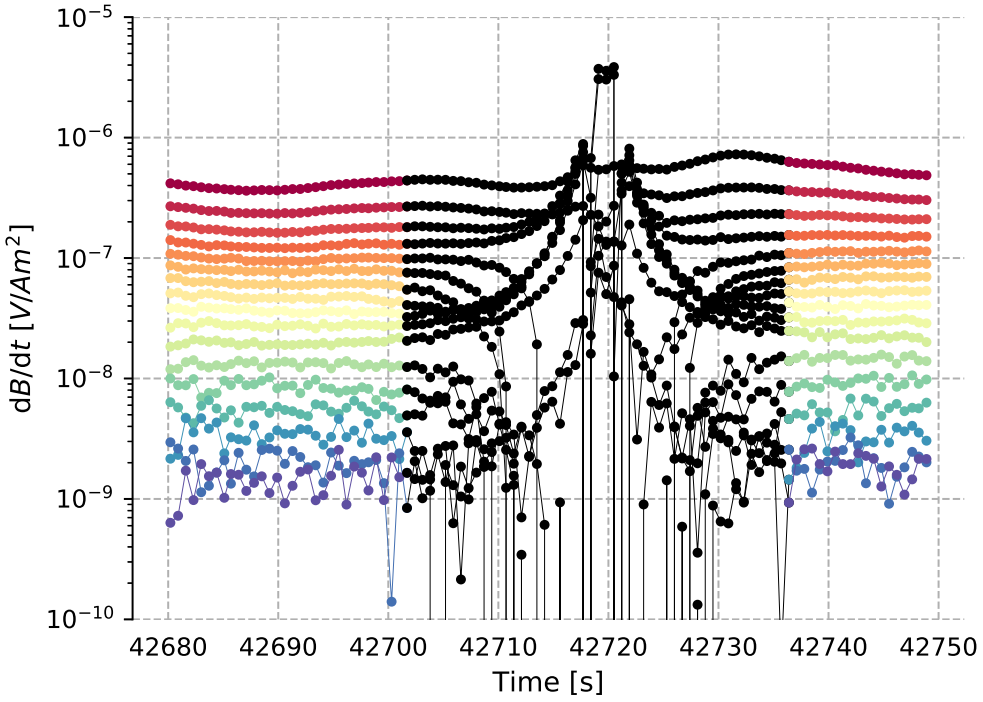
In figure 3.12b an example of a reconstruction of the input segment in figure 3.12a is shown. It is clearly seen that the autoencoder is able to reconstruct non coupled samples quite well, while coupled samples are reconstructed poorly as desired. The output in the coupled segment does not have similar dB/dt values to the input and thus will result in a much larger error than the non coupled segment.

Precision and recall of the autoencoder from a five fold cross validation is presented in table 3.6. The autoencoder perform on par with the 1D CNN, with standard deviations still suffering from a great variance in the training and validation set. In the ROC curves in figure 3.13, a variance in the performance of each iteration is observed as was the case with the previous classifiers. The threshold independent AUC score is slightly better at 96 (3.0). The key aspect to note however, is that the results are achieved utilizing only the non coupled class.

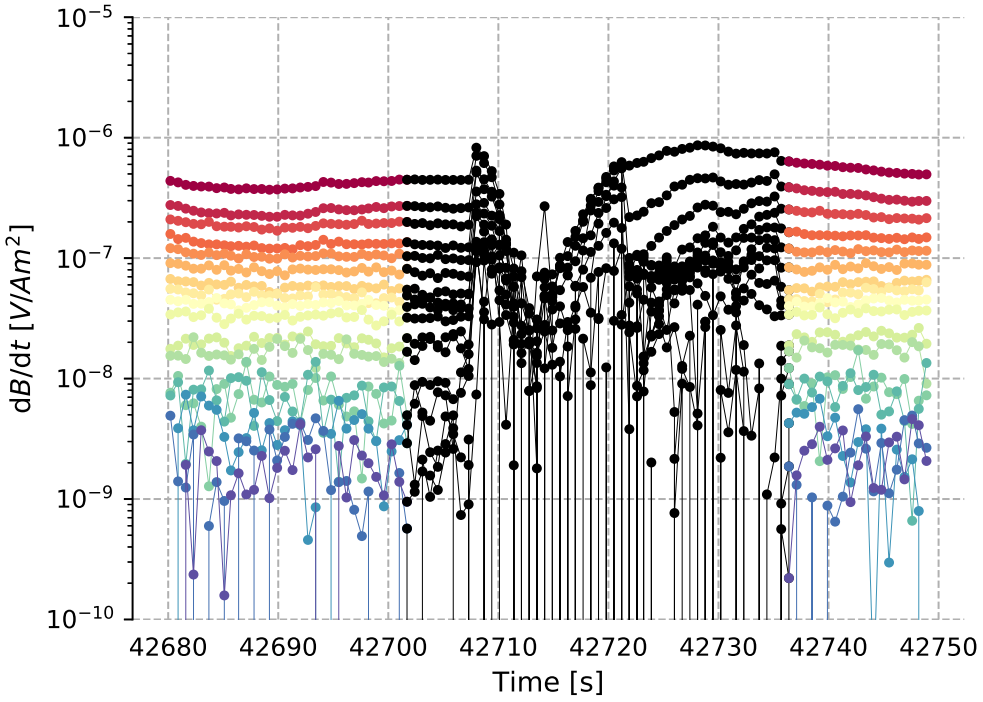
In figure 3.14 a histogram of the mean absolute error between the target and prediction of the autoencoder is shown. Errors larger than 1 exist but are few and are not shown to keep the plot readable. Especially iteration 5 has a bit of a tail

TABLE 3.5: Autoencoder layout.

Layer	Neurons	Activation
Input	340	Linear
Fully connected	170	ReLU
Fully connected	85	ReLU
Fully connected	170	ReLU
Output	340	Linear



(A) Input to the autoencoder.



(B) Reconstructed output of the autoencoder.

FIGURE 3.12: Autoencoder reconstruction.

TABLE 3.6: Autoencoder precision and recall from a five fold cross validation.

	Precision	Recall	Samples
Coupled	69 (17)	90 (7.4)	1175 (752)
Non coupled	97 (2.1)	89 (10)	4115 (752)

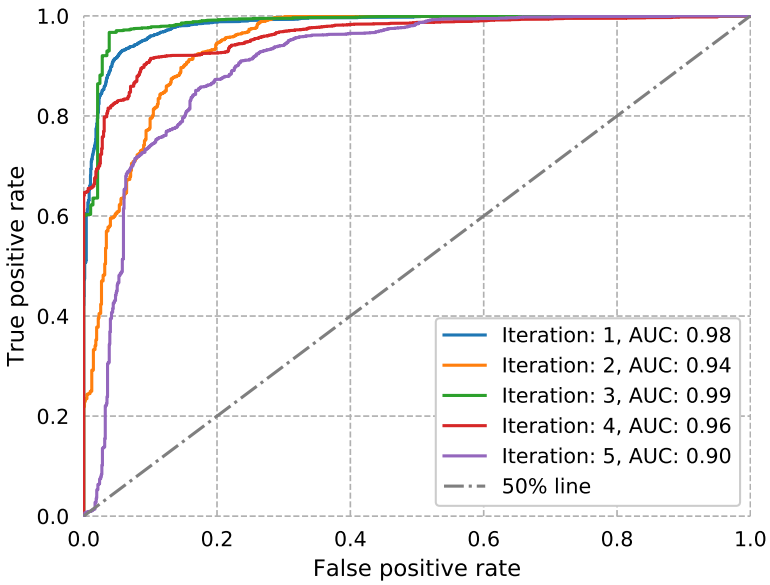


FIGURE 3.13: ROC curves of the autoencoder.

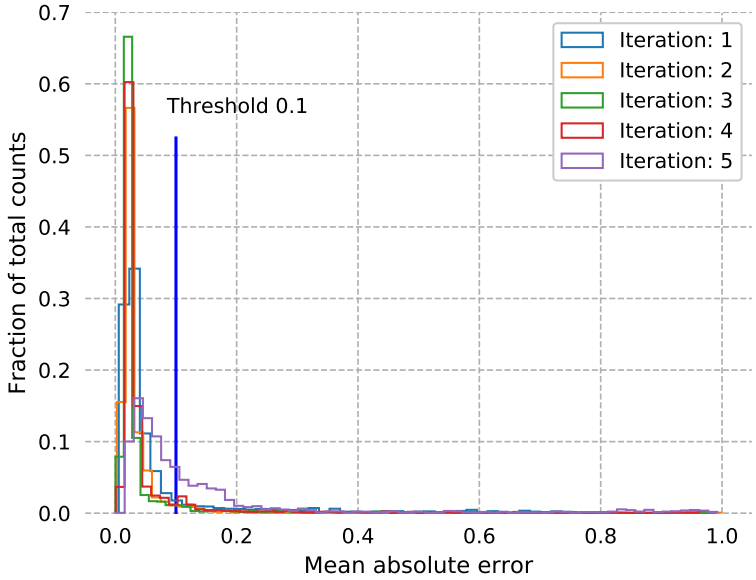


FIGURE 3.14: Distribution of mean absolute error of autoencoder predictions.

with uncertain predictions, otherwise most predictions are gathered in the same bin towards zero as expected for the non coupled class.

The misclassification primarily occur in the transition region, a problem common to all three classifiers. An example is found in figure 3.15.

As with the random forest model, the autoencoder is tested for convergence as a function of training set size. In figure 3.16 the total number of errors made by the autoencoder when trained on an increasing training set size is shown. The training and validation set is equivalent to that of iteration 5. The training set size is incrementally increased from 1% to 100% of the total training set size, starting from the beginning. Comparing with the learning curve of the random forest in figure 3.5 they are very similar. The autoencoder however, is already learning from the beginning where the random forest was uncertain. Likely this happens since the autoencoder only need to learn the non coupled class and the entirety of the training set consist of the non coupled class, whereas the random forest had to learn two different classes from a small training set with an unbalanced distribution. The autoencoder is still not converged and could benefit from more training data.

Though results are not at an operational requirement, they are the best so far on par with the CNN and more excitingly achieved with a model trained only on the non coupled samples. This opens up for new possibilities that could make training easier and less susceptible to human annotation errors, as will be discussed in chapter 4.

3.6 Test Set Results

For a final evaluation of the three models, they are trained on the entire training set and then tested on the unseen test set. Accuracy, recall and precision can be found in table 3.7. Since there is only one test set we cannot compute a measure of uncertainty. The models do entail a level of randomness in parameter initialization and stochastic

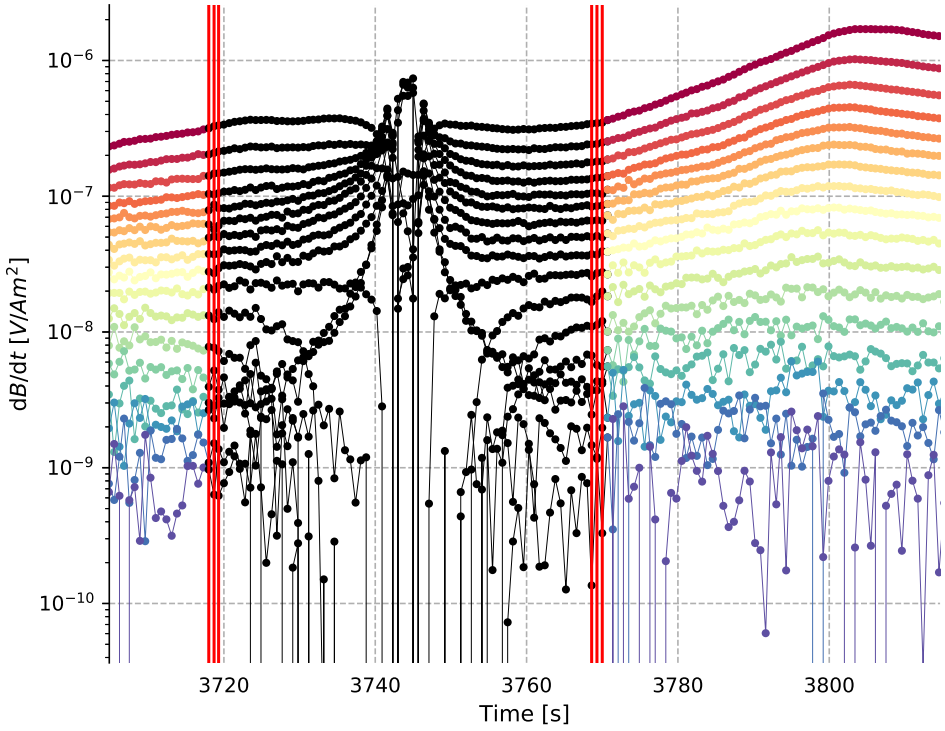


FIGURE 3.15: A segment of data classified by the autoencoder. Each colored line is a gate. Black dots are gate values manually marked as coupled. Vertical red lines are soundings incorrectly classified by the autoencoder, all soundings without vertical red lines have been correctly classified.

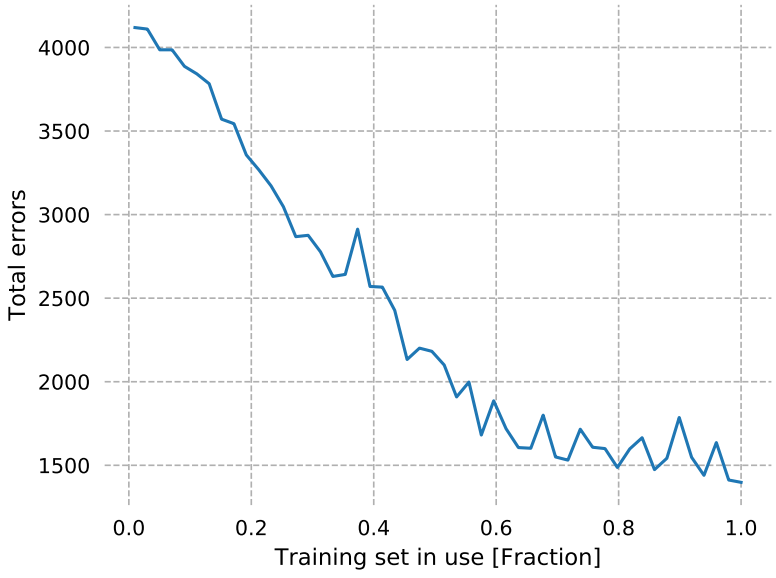


FIGURE 3.16: A learning curve of the autoencoder, showing the decrease in classification error as a function of increasing training set size.

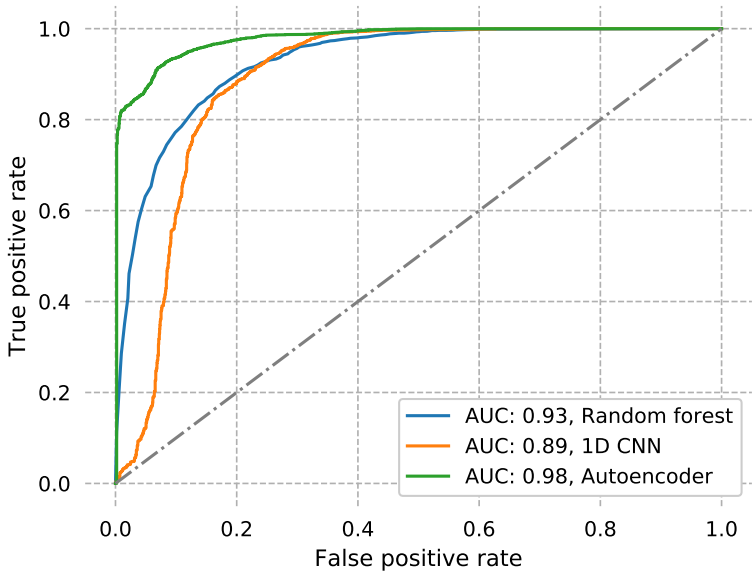


FIGURE 3.17: ROC curves of the predictions on the test set.

gradient descent, thus if we retrain the network, the output will be slightly different. However, this uncertainty is insignificant and therefore not reported. In figure 3.17

TABLE 3.7: Accuracy-, recall- and AUC score for all evaluated models on the test set.

	Accuracy	Recall coupled	Recall non coupled	AUC
Random forest	83	87	80	93
1D CNN	88	84	93	92
Autoencoder	**89**	**95**	**86**	**98**

the ROC curves of each model on the test set is shown. Given the random forest simplicity and ease of use, its result are impressive, performing on par with the CNN on the test set. The CNN was expected to outperform the random forest based on the cross validation results, which is not the case on the test set. To some extent this can be attributed to a by-product of the current windowing method, as it does not account for holes in the data as further discussed in Chapter 4. Performance of the autoencoder on the test set is superior with a sufficient recall of the coupled class, although a higher recall on the non coupled class is still needed. The AUC of the autoencoder is almost perfect with a 98% chance of ranking a non coupled sample higher than a coupled, resulting in a relatively low trade off between true- and false positives. The accuracy is on level with the CNN, but as stated in section 2.7 the accuracy is as bad measure in case of imbalance. The autoencoder achieve a very high recall of the coupled class at the expense of the non coupled class, a wanted behaviour, though this entails a lower accuracy score, since coupled samples are outnumbered 1 to 3.

Chapter 4

Discussion

The goal of the thesis is to investigate machine learning based methods for detection of couplings in tTEM data. Detecting and culling couplings in tTEM data presents one of the most time consuming steps of the tTEM processing cycle. Efficient solutions to this problem is becoming a necessity as the system is increasingly used. Three different approaches to the problem were investigated. Promising results were achieved and possibilities for future research directions were identified along with some limitations of the method.

A big problem with all three models evaluated is the large standard deviation observed from the cross validation. It was found that the deviation can be explained by a very differing geology in the survey area and an uneven distribution of coupled samples in the training- and validation sets. Since the data is sequentially measured in discrete intervals as the equipment moves forward in space, there is an inherent selection bias, in that some geology might not be present in both the training and validation set. This can be addressed by sampling the training-, validation-, and test set randomly across the entire data set. However, in doing so the contextual information is lost, an unwanted effect since couplings usually affect more than just one sounding, further [6], [8], [11] conclude that context is necessary for detection of couplings. Thus the evaluation is suffering from selection bias in the splitting of the data, it could likely be reduced by obtaining a larger data set. In retrospective we could have selected a survey area with a more uniform geology to make for an easier starting point, though the problem of varying geology is still very real and will have to be addressed at some point.

A disadvantage of the random forest model, is that it takes as input a single sample at a time and output a prediction for that sample. As a result, single soundings could receive a classification that did not correspond to its surroundings as illustrated in figure 3.6. From a domain perspective, it is very unlikely for a single sounding not to share the class of its neighbours. A possible solution could be inspired from canny edge detection as employed in computer vision . As a post processing step, all samples with a weak classification score could be iterated through to check if its two neighbours have a strong classification of the same type and if so change the classification of the weakly classified sample accordingly. However, we instead adopt a window sampling technique to improve the context of the input. This allow classification of a sounding to be based on its surroundings. Naturally there is a limit to how big a window is needed. Too big a window would result in finer details being lost. Conversely, a small window would miss larger scale details, a problem similar to that of scale invariance in computer vision tasks. The size and overlap of the window is found by a manual search based on the cross validation.

To fully utilize the larger input space and context resulting from the data windows, a 1D CNN was developed and tested. Although the results improved upon those achieved with the random forest model in the cross validation, evaluation on

the final test set were slightly worse. It could be a result of the CNN overfitting the training data and therefore doing worse on the unseen test data. However, the problem of single soundings being misclassified in opposition to the neighbourhood is diminished.

From the classification decisions made by both the random forest and the 1D CNN, it is apparent that identifying a location with a strong coupling is not problematic. The majority of the disagreement between the manually labelled classifications and the ones predicted by the automated models occur in the transition from non coupled to coupled segments. It is not surprising since a coupling increase gradually in amplitude as the distance to the coupled object decreases. The result being that it is impossible to exactly state at which sounding a coupling starts to be evident and at which it stops. The labelling of couplings in data is therefore highly dependent on the experience and safety concerns of the operator. It questions the evaluation method, which is purely based on the classifiers agreement with manually annotated data. If the goal is to hit a recall or accuracy of 100% for example, we have to assume that the manual annotations are 100% correct. In the case of images, there is often not much doubt whether an image depicts a cat or a dog, but in the case of annotating couplings there will be an uncertainty in the border placement between coupled and non coupled segments, as the transition is simply not sharply defined. Annotating labels specifically for the purpose of training a machine learning model might increase the performance metrics, but does not address the underlying problem that there is a margin of uncertainty when deciding the transition.

Attempting to address the problem with high variation in the results of the random forest and 1D CNN and partially the problem with uncertain labelling of the transition regions, an unsupervised autoencoder were developed and tested. The autoencoder was only trained on the non coupled segments and thus the manual annotations of the data did not directly influence training of the model, neither did the class imbalance in the training set.

A number of potential benefits are implied when using a single class classification strategy. Any classification model can only be as good as the data it is trained on. In the supervised case, if an uncertainty in the labelling of data is introduced, it will affect the certainty of the model. As argued earlier, the supervised labelling of tTEM data is not necessarily consistent, and many soundings marked as coupled might just be so because they are close to a coupling and the operator wants to be completely sure that all effects of the sounding are removed. From a data perspective this means that some of the soundings marked as coupled may look identical to a non coupled sounding, which degrade the quality of the training set. In the unsupervised case, given enough data we can just remove the coupled segments with a very low tolerance for couplings, ensuring that any sign of a coupling is removed from the training set. Naturally this does not fix the problem of evaluation against the supervised labelling.

Results of the autoencoder greatly improved on the two other models. The majority of misclassified samples were observed to be in the transition region as previously, but that was expected given the discussion above.

The improved results of the autoencoder suggest that viewing the problem as anomaly detection, with couplings as anomalies, instead of a binary classification could be desirable. One could reasonably question whether learning the coupled class and viewing the non coupled class as anomalies could be more productive. A reason to choose the coupled class as anomalies is that the non coupled class can be simulated very well, leading to another potential benefit of the one class classification strategy.

Andersen et al. [6] conclude that their network need to be adapted to the local survey and geological conditions, suggesting that a small subset of a survey is manually inspected and used as training set, after which the network can be applied to the remaining part of the data set. A similar trend of training sets lacking general representative capability is noted. However, we aim to fully automate the process of culling data. This is potentially made possible by the autoencoder.

Given a layered model of the earth, what is known as a forward response can be calculated, which is equivalent to a measured sounding. A layered model of the earth, is a model that describe the surface based on a number of stacked layers, their thickness and resistivity. Theoretically, all possible earth layer configurations in an area, e.g. all of Denmark, can be approximated by creating a layered model and sweeping over layer depth and layer resistivity.

By training a one class model on such a data set, it is possible that appropriate generalization can be achieved. This would make every step of the processing line fully automated with no need for physical measurements. However, it involves an enormous amount of data, spanning a broad range of different geologies. The variance of the synthetic training set would be large, forcing the autoencoder to learn a more general function, which could pose a problem in separating the coupled and non coupled class. Furthermore, if we are to use the context of a sounding for classification, one would have to not just create independent synthetic forward responses, but a continuous model. Unfortunately it was not possible to investigate this idea within the limited time span of the thesis work.

A source of error for both the CNN and autoencoder is the windowed sampling technique. It does not account for holes in the data, stemming from the tTEM system turning a corner, or for other reasons being shortly turned off. The effect is that it groups together soundings that are not neighbours in the spatial domain. This might lead to a drastic and sudden change in dB/dt values from one sounding to the next, because the hole is not accounted for, essentially leading to an anomaly in the data. Because soundings are always recorded in intervals of approximately 0.7 seconds, a solution could be to fill in the holes with artificial soundings all having dB/dt values of zero. In this way the artificial soundings have no influence, as they turn off the neurons to which they are input. Another solution is to only sample a sequence until a hole is encountered and then start the window from the beginning of the next sequence, such that the window 'jump' over holes.

The hyperparameters, though relatively few, were all manually adjusted based on improving the cross validation scores. Other more exhaustive types of hyperparameter optimization techniques exist such as grid search, random search or more advanced evolutionary optimization techniques. These can be very time consuming and computationally expensive and are mostly used for fine tuning the hyperparameters, leading to small gains in performance. Since we did not arrive at a production quality model, manual hyperparameter optimization was deemed satisfactory.

The analysis and conclusions of the report is limited by the relatively small survey area of about $2\,\mathrm{km}^2$. To draw more general conclusions, experiments on a broad range of survey areas would have to be conducted. However that was not feasible due to time constraints.

Chapter 5

Outlook

From the findings of this thesis a number of possibilities for future work is suggested. A one class classification strategy, which according to the best of my knowledge has not attempted before on TEM data, seems to be a promising method for rejection of coupled data. It can avoid some of the difficulties regarding labelling of coupled data points. It would be especially interesting to investigate whether a one class classifier could be successfully trained using only synthetically created data points. Other models and strategies can possibly be inspired from the field of anomaly detection, specifically tasked with one class classification.

Investigating the possibility and potential of including more features could also be beneficial. Other features could include GPS coordinates and a map to infer distances from a road or buildings etc.

Currently validation against the manually annotated labels pose a problem for performance metrics, as there is an inherent uncertainty in the labelling of couplings as discussed in chapter 4. An alternative strategy could be to use manual annotations for testing and tuning a network model. Then in the final evaluation of the model, we could let an expert look through the predictions of the model and annotate all predictions that the expert find incorrect, as opposed to validating against a predefined annotation.

In this regard, it would be especially helpful to define a common set of metrics for measuring the performance of models. When it comes to classification of imbalanced data sets, there are no standard adopted metrics, but e.g. [23] suggest a set of metrics like recall, precision, AUC and ROC curve.

Chapter 6

Conclusion

This thesis aimed to identify effective machine learning based methods for rejection of couplings in tTEM data. Currently the data is manually inspected for couplings by experts in the field, a very time consuming process comprising up to 50% of the data processing cost.

Three different methods were developed and evaluated, a random forest, a one dimensional convolutional neural network and an autoencoder. From evaluation of the three models, various problems and solutions to detection of couplings were identified. Especially promising results were achieved using an unsupervised autoencoder neural network. The network only utilize non coupled soundings in order to learn a nonlinear function, encoding the input space in a lower dimensional representation and from there reconstructing the input via a learned decoding function. Reconstructions of coupled soundings result in a large error on the output, as the network has only learned to reconstruct the non coupled soundings. The autoencoder successfully detect $\sim$95% of the coupled soundings and $\sim$86% of the non coupled soundings.

The study is based on a data set consisting of 33000 soundings collected with the tTEM system in the area of Stendalmark, Denmark. The results are limited by the relative small size of the data set. The survey area covers $\sim 2\,\mathrm{km}^2$ and as such represent a limited type of geology, although the geology within the area was found to vary a lot.

The results as is, are not at a stage where they can be directly implemented in existing processing software such as Aarhus Workbench [29], however they lay the foundation for further research into machine learning based approaches for detecting couplings based on one class, whereas other approaches so far have been based on binary classification. It is possible to create high quality simulations of non coupled soundings, thus there is a great prospect of fully automating the TEM data processing line, with a machine learning model purely trained on synthetic non coupled soundings.

Bibliography

[1] Misac N Nabighian and James C Macnae. "Time Domain Electromagnetic Prospecting Methods". In: *Electromagnetic Methods in Applied Geophysics: Volume 2, Application, Parts A and B*. 2012. Chap. 6, pp. 427–520.

[2] Paul A. Bedrosian, Cyril Schamper, and Esben Auken. "A comparison of helicopter-borne electromagnetic systems for hydrogeologic studies". In: *Geophysical Prospecting* 64.1 (Jan. 2016), pp. 192–215.

[3] Esben Auken et al. "tTEM — A towed transient electromagnetic system for detailed 3D imaging of the top 70 m of the subsurface". In: *GEOPHYSICS* 1 (2019), E13–E22.

[4] K I Sorensen and Esben Auken. "SkyTEM–a New High-resolution Helicopter Transient Electromagnetic System". In: *Exploration Geophysics* 35.3 (2004), pp. 194–202.

[5] Kurt I. Sørensen, A.V. Christiansen, and Esben Auken. "4.4 The transient electromagnetic method (TEM)". In: *Groundwater Resources in Buried Valleys, a Challenge for Geosciences* (2006), pp. 65–76.

[6] Kristoffer K. Andersen et al. "Artificial neural networks for removal of couplings in airborne transient electromagnetic data". In: *Geophysical Prospecting* 64.3 (2015), pp. 741–752.

[7] Marcus F. Flis, Gregory A. Newman, and Gerald W. Hohmann. "Induced-polarization effects in time-domain electromagnetic measurements". In: *GEOPHYSICS* 54 (May 1989), pp. 514–523.

[8] Esben Auken et al. "An integrated processing scheme for high-resolution airborne electromagnetic surveys, the SkyTEM system". In: *Exploration Geophysics* 40.2 (2009), pp. 184–192.

[9] Hydro Geophysics Group. URL: `http://hgg.au.dk/`.

[10] HGG. *Guide for processing and inversion of SkyTEM data in the Aarhus Workbench*. Tech. rep. Department of Earth Sciences Aarhus University, 2011. URL: `http://www.hgg.geo.au.dk/rapporter/guide_skytem_proc_inv.pdf`.

[11] P.-A. Reninger et al. "Singular value decomposition as a denoising tool for airborne time domain electromagnetic data". In: *Journal of Applied Geophysics* 75.2 (Oct. 2011), pp. 264–276.

[12] Tin Kam Ho. "Random decision forests". In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. IEEE Comput. Soc. Press, 1995, pp. 278–282.

[13] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32.

[14] Web of Science. *Random Forest*. 2019. URL: `http://apps.webofknowledge.com`.

[15] Lior Rokach. "Ensemble-based classifiers". In: *Artificial Intelligence Review* 33.1-2 (Feb. 2010), pp. 1–39.

[16] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. 2nd ed. Springer, 2009.

[17] Christopher M Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[19] William Wei Hsieh. *Machine learning methods in the environmental sciences : neural networks and kernels*. Cambridge University Press, 2009, p. 349. ISBN: 9780511627217.

[20] Jonathan Masci et al. "Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction". In: Springer, Berlin, Heidelberg, 2011, pp. 52–59.

[21] Diederik Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations* (May 2014).

[22] Sotiris Kotsiantis, Dimitris Kanellopoulos, and P Pintelas. "Data Preprocessing for Supervised Learning". In: *International Journal of Computer Science* 1 (May 2006), pp. 111–117.

[23] Haibo He and E A Garcia. "Learning from Imbalanced Data". In: *Knowledge and Data Engineering, IEEE Transactions on* 21 (May 2009), pp. 1263–1284.

[24] Tom Fawcett. "An introduction to ROC analysis". In: *Pattern Recognition Letters* 27.8 (June 2006), pp. 861–874.

[25] Chollet François et al. *Keras*. https://keras.io. 2015.

[26] Rami Al-Rfou et al. "Theano: A Python framework for fast computation of mathematical expressions". In: *arXiv e-prints* abs/1605.02688 (May 2016). URL: `http://arxiv.org/abs/1605.02688`.

[27] Toke Frederiksen. *Github repository*. 2019. URL: `https://github.com/TokeF/MasterThesis`.

[28] Philipp Probst, Anne-Laure Boulesteix, and Marvin Wright. *Hyperparameters and Tuning Strategies for Random Forest*. May 2018.

[29] HydroGeophysics Group. *Aarhus Workbench*. 2018. URL: `http://hgg.au.dk/software/aarhus-workbench/`.