Matthäus Lenz, BSc

Machine Learning in Network Protection

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

Masterstudium Elektrotechnik-Wirtschaft

eingereicht an der

Technischen Universität Graz

Betreuer

Univ.-Prof. DDipl.-Ing. Dr.techn. Robert Schürhuber

Institute of Electrical Power Systems

Co-Betreuer
Dipl.-Ing Alexander Rainer

Graz, September 2020

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Datum

Unterschrift

Institute of Electrical Power Systems

Machine Learning in Network Protection

A master thesis by
Matthäus Lenz

Supervisor
Dipl.-Ing Alexander Rainer
Univ.-Prof. DDipl.-Ing. Dr.techn. Robert Schürhuber
Em.Univ.-Prof. Dipl.-Ing. Dr.techn. Lothar Fickert
Dr. Andreas Theil

September 2020

Graz University of Technology
Institute of Electric Power Systems
Inffeldgasse 18/I
8010 Graz
Austria

Head of Institute
Robert Schürhuber

Supervisor
Dipl.-Ing Alexander Rainer
Univ.-Prof. DDipl.-Ing. Dr.techn. Robert Schürhuber
Em.Univ.-Prof. Dipl.-Ing. Dr.techn. Lothar Fickert
Dr. Andreas Theil

A master thesis by
Matthäus Lenz

September 2020

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz,

TBA

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am

Name Student

Abstract

Machine learning is already being used in various areas and will open many other opportunities in the future to optimize grid systems and operations. This master's thesis investigates how machine learning can be used in network protection by looking at two examples.

On the one hand, it is examined whether it is possible to use machine learning to classify different types of short circuits and earth faults. Since earth fault locating methods currently used in compensated medium voltage networks only detect the earth-faulted branch in a substation but do not allow the earth fault to be localized, problems in troubleshooting and isolation errors can occur due to higher voltage stresses. On the other hand, attempts are made to localize single-pole earth faults in medium voltage networks via machine learning.

The present work deals with the machine learning algorithm of neural networks. In order to generate data for the training and testing of neural networks, short circuits and earth faults are simulated in various medium voltage networks in the DIgSILENT PowerFactory software.

The results show that neural networks are well suited for fault classifications. Regarding the earth fault locating task, the outcomes indicate that neural networks can be a useful tool to locally limit earth faults. A combination of classical earth fault-locating methods with earth fault-locating neural networks can increase the accuracy and reliability of both methods.

Kurzfassung

Machine Learning wird bereits in unterschiedlichen Bereichen angewandt und zukünftig viele weitere Chancen eröffnen, um Netzsysteme und den Betrieb von Netzen zu optimieren. Wie Machine Learning im Bereich des Netzschutzes angewandt werden kann, wird in dieser Masterarbeit anhand von zwei Beispielen erläutert.

Zum einen wird untersucht, ob es möglich ist, mit Hilfe von Machine Learning verschiedene Arten von Kurzschluss- und Erdschlussfehlern zu klassifizieren. Da derzeit eingesetzte Erdschlussortungsverfahren in kompensierten Mittelspannungsnetzen nur den erdschlussbehafteten Abgang im Umspannwerk erfassen, jedoch keine örtliche Eingrenzung des Erdschlusses ermöglichen, kann es infolge zu Problemen in der Fehlersuche und, aufgrund höherer Spannungsbeanspruchungen, zu Isolationsfehlern kommen. Aus diesem Grund wird mit Hilfe von Machine Learning in Mittelspannungsnetzen einpolige Erdschlussfehler örtlich einzugrenzen.

Die vorliegende Arbeit beschäftigt sich mit dem Machine Learning Algorithmus der neuronalen Netze. Um Daten für das Trainieren und Testen von neuronalen Netzen zu generieren, wurden Kurzschluss- und Erdschlussfehler in verschiedene Mittelspannungsnetze in der Software DIgSILENT PowerFactory simuliert.

Aus den Ergebnissen lässt sich schließen, dass neuronale Netze zu Fehlerklassifikationen geeignet sind. Hinsichtlich der Erdschlussortung zeigt sich, dass neuronale Netze ein nützliches Werkzeug sein können, um Erdschlüsse lokal zu begrenzen. Eine Kombination von klassischen Erdschlussortungsverfahren und erdschlussortenten neuronalen Netzen kann die Genauigkeit und Zuverlässigkeit beider Methoden erhöhen.

List of Symbols

I_{CE}	Capacitive earth fault current
I_0	Zero-sequence current
U_n	Nominal voltage
C_E	Line-to-ground capacitance
X_P	Reactive impedance of the arc-suppression coil
L_P	Inductance of the arc-suppression coil
I_L	Inductance current
d	Damping factor
I_{act}	Active residual current
u_0	Zero-sequence voltage
CSV	Comma-separated values
SCL	Short-circuit level
$\underline{Z}_{Grid}$	Grid impedance
c	Voltage factor
$\underline{Z}_{Tr}$	Transformer impedance
uk	Short-circuit voltage
$\underline{Z}_{Load}$	Load impedance
S	Apparent power
$\underline{Z}_{Cable}$	Cable impedance
l	length
R'	Reactive Impedance per km
X'_L	Reactive Impedance per km
km	Kilometre
$\mathrm{I''_k}$	Absolut short circuit current
$g()$	Activation function
w	Weight

b	Bias
α	Learning rate
Y	Expected output (target) of a neural network
Y'	Output of a neural network

Table of Contents

1 Introduction

Applications in the field of artificial neural networks, commonly known as artificial intelligence, are rapidly increasing. In the future, artificial intelligence will significantly increase the possibilities in the field of electrical engineering. Optimizing efficiency in the energy sector and keeping grid systems stable are just two prospects out of an endless number of potential applications. These opportunities of neural networks can be applied in the field of network protection.

1.1 Motivation

Almost all medium voltage networks in Austria and Germany are operated as compensated networks. The detection and localization of earth faults in compensated networks is a complex task. Currently, measuring devices cannot precisely locate the positions of earth faults; rather, the faulty branch as a whole is detected. In compensated grids, earth fault currents are low, and a persisting earth fault allows the grid to continue operating. Finding such earth faults along branches with distances of up to 20 km can require several hours of work. However, the longer an earth fault remains in a compensated network, the higher the isolation stress in the healthy feeders. Therefore, there is a lot of interest in more exact earth fault-locating systems.

In the field of computer vision, machine learning has made massive steps forward in the last recent years. Machine learning algorithms can discover patterns in large amounts of data where humans can no longer keep up with. It is possible to extract information from data and then analyze and manipulate this information in different ways. By doing so, machine learning could open new approaches in the field of network protection, for example, by classifying faults and narrowing earth faults in medium voltage networks.

1.2 Scope of the Work

The aim of this master's thesis is to investigate possible machine learning applications in the field of network protection. It starts by providing an overview of the theoretical basis. This is followed by an introduction to commonly used earth fault-detecting methods.
First, this thesis examines whether neural networks are able to classify different kinds of faults correctly. Various faults in an exemplary medium voltage network were simulated with the use of the software DIgSILENT PowerFactory. The generated and automatically labeled data is then used to train and further test a neural network for fault classification.
Second, it was investigated whether neural networks can be a useful tool in the earth-fault-locating issue by narrowing the earth-faulted section along a branch of a medium voltage cable network.
In this approach, single-pole earth faults were simulated along the cables while using different exemplary medium voltage cable networks. The generated and labeled data was then used to train and evaluate neural networks for narrowing earth faults.

In the end, it was examined how such an earth-fault-locating neural network can be implemented in operating systems.

1.3 Research Questions

1. Can neural networks be applied in the area of network protection?
2. Can neural networks be used to distinguish faults in medium voltage networks?
3. Can neural networks be a useful tool to narrow earth fault positions in medium voltage networks?
4. Which data is needed to train and apply such neural networks successfully?
5. Which aspects of medium voltage network topologies influence the accuracy of neural networks?
6. How can an earth-fault-locating neural network be implemented in operating systems?

2 Short Circuits and Earth Faults

Symmetric and asymmetric faults in power grids are caused by isolation breaks of the dielectric or by line breaks. If the isolation between the conductors of a three-phase system fails due to aging, operational or atmospheric overvoltage, an arc occurs at the fault location. Which practically short-circuits the consumer impedances. Possible faults in power grids are defined as in Figure 2-1. [1] [2]

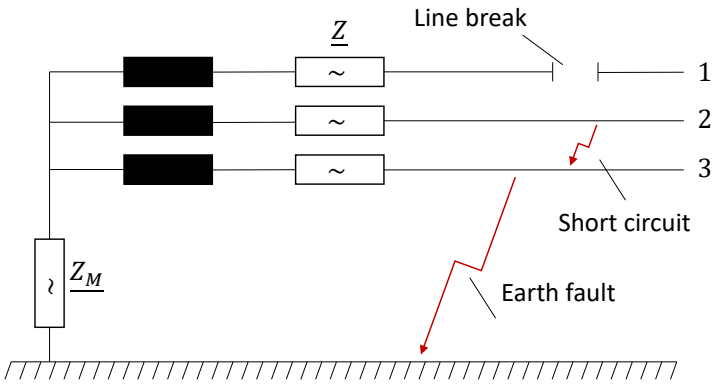


Figure 2-1, Definition of faults in power grids [1]

A **short circuit** is a resistance-free or low resistance connection between one or more active components with nominal voltage. [1]

An **earth fault** is a resistance-free or low resistance connection between live parts and earth if the neutral treatment is insulated or resonant grounded. [1]

A **line break** occurs if the line disconnects through breaks in lines, cables, transformers, or dysfunctional circuit breakers. [1]

2.1 Classification of Faults

Short circuits can reach ten times the nominal current in a system. Thus, short circuits account for the most dangerous faults in electric power grids. Basically, short circuits and earth faults are distinguished into five categories, as described in [3]:

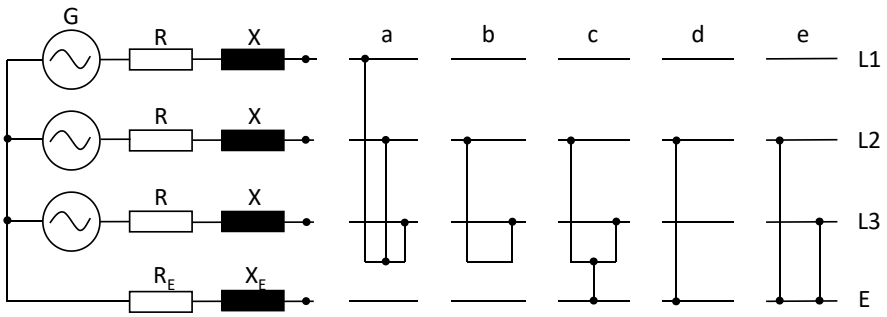


Figure 2-2, Different kinds of short circuits and earth faults. [4][3]

a) 3-phase short circuit

A 3-phase short circuit occurs between all three conductors. This kind of short circuit puts the most significant strain on a network and on switching devices. A 3-phase short circuit is the last stage of a developing fault. A developing fault often starts with a single-pole earth fault. Through arc extension and the increase in voltage in the healthy phases, a single-pole earth fault develops into a 3-phase short circuit. In a 3-phase short circuit, all currents have the same magnitude. The voltages decrease symmetrically in all phases, from the transformer to the fault location [4].

b) Double-phase short circuit without ground

A double-phase short circuit occurs between two conductors. In both affected phases, a short-circuit current flows. This kind of short circuit is very rare and sometimes appears when two oscillating overhead lines touch each other [4].

c) Double-phase short circuit with ground

A double-phase short circuit occurs between two conductors and the ground. In the case of a double-phase short circuit with ground, only the voltages in the phases of the short circuit decrease [4].

d) Single-pole earth fault

An earth fault can only appear in a three-phase power system. In case of an earth fault, one conductor has contact with ground or the neutral point. The equalizing current of the conductor to earth increases the earth potential at the fault location. It lowers the potential of the faulty conductor. By connecting the conductors in the three-phase network, the total voltage of the other two healthy conductors is increased by the amount. This can lead to voltage flashovers on insufficiently stress-resistant components. [4]

e) Double-phase short circuit with ground and separated fault points

The typical consequence of insufficient isolation after a single-pole earth fault. A single-pole earth fault becomes a double earth fault due to the excessive voltage stress in networks with an isolated or a resonant neutral point treatment [5].

2.2 Neutral Point Treatment and Protection

The neutral point treatment of transformers in a power grid has a significant influence on how a network is responding to earth faults, in particular with the common single-pole earth fault in medium voltage networks. The neutral point treatment of power grids is distinguished in:

- Networks with an insulated neutral
- Networks with a resonant grounded neutral
- Networks with a low impedance neutral point [1]

Neutral treatments have an impact on the following quantities:

- The magnitude of the earth fault current (single-pole or double-pole)

- Step and touch voltage
- Voltage stress of the healthy feeders in case of failures
- The voltage stress on the first extinguishing switch pole in a three-pole switch-off
- The necessity to switch off defective equipment to avoid supply interruptions [1]

The neutral point treatment is defined as a galvanically connected network area. Network areas are limited by a transformer with galvanic isolation and/or through open circuit breakers. Within a network, areas with mixed forms of low impedance and compensated neutral point treatments cannot be used [1].

Table 2-1 shows the distribution of different neutral point treatments in different voltage areas in Germany, as in [1].

Voltage	Neutral point treatment		
	Insulated neutral	Resonant grounded neutral	Solidly of low impedance grounded neutral
10 kV	14,30%	72,60%	13,10%
20 kV	0%	100%	0%
110 kV	0%	91,10%	8,90%
220 kV	0%	0%	100%
220 kV	0%	0%	100%
380 kV	0%	0%	100%

Table 2-1, Type of neutral point treatment in Germany [1]

2.2.1 Insulated Neutral Point

Medium voltage networks with low expansion and a low capacitive earth fault current I_{CE} can be operated with an insulated neutral point. An arc in transmission line networks will be self-extinguishing if the capacitive earth fault current does not overrun a certain amount of amps. [3]

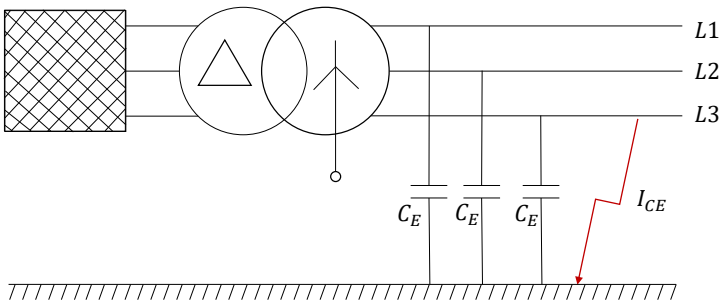


Figure 2-3. Single-pole earth fault in a network with insulated neutral [3]

The capacitive earth fault current in Figure 2-3 is approximated using the following equation, as in [3]:

$$I_{CE} = 3I_0 = \frac{\sqrt{3}U_n}{|Z_0|} = \sqrt{3}\omega C_E U_n \qquad 2\text{-}1$$

In the case of single-pole earth fault in phase $L1$, the current flows over the line-to-ground capacitance of the phases $L2$ and $L3$ to the earth fault position. The current over the line-to-ground capacitance of $L1$ discharges to zero. [1]

The main advantage of the insulated treatment of the transformers is that the line-to-ground capacitance is limiting the earth fault current. If the medium voltage network is not too extensive, meaning the line-to-ground capacitance is small enough, the current will be below the tripping limit of the protection system. [1]

The disadvantage of an insulated neutral point treatment is a voltage increase by the factor $\sqrt{3}$ of the phases, which are not affected by the fault. This can, especially with previously damaged equipment, lead to another earth fault, so that the standing earth fault changes into a double earth short circuit. [1]

2.2.2 Resonant Grounded Neutral (Compensated Network)

The more extensive a network is, the larger the capacitive earth fault current is. Through an arc-suppression coil (ASC), also called a Petersen coil, the earth fault current is limited. In compensated networks, one or more neutral points of transformers are grounded with an arc-suppression coil. The capacitive current over the line-to-ground capacities of the power grid is superimposed with the inductive current from the arc-suppression coil. The earth fault current is lower than in an insulated network. [1]

At a matching of the inductance of the ASC to the capacitance $C_E = C_0$ of the power grid, in case of an earth fault, only the residual earth fault current $I_{Residual}$ flows at the fault location. The residual earth fault current consists of an active component (residual wattage current) and the harmonic components [3]. The deletion limit for the residual current is 60 A for networks ≤20 kV and increases for 110 kV networks to 130 A. [6]

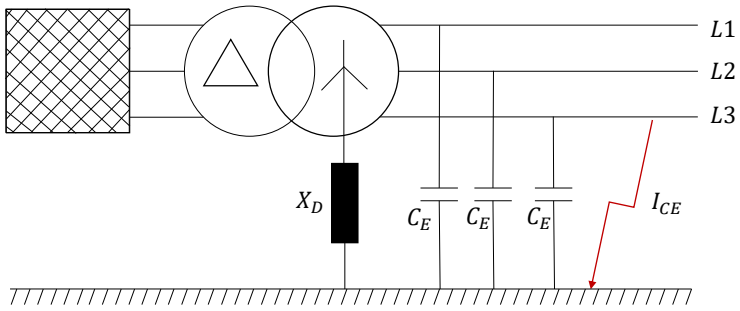


Figure 2-4, Single-pole earth fault in a compensated network [3]

As in insulated networks, the tensions between the healthy phases and the earth increase by a factor of $\sqrt{3}$, during an earth fault. [6]

When the compensation condition (resonance condition) is met, the amount of inductive current via the ARC is the same as that capacitive electricity over the earth capacities of the network area. The inductivity of the coil is determined through the following equation [1]:

$$X_D = \frac{1}{3\omega C_E} \qquad 2\text{-}2$$

The main advantage of the compensated network is the low earth fault current. Even in extensive networks with a high earth capacity, the protection system will not trigger in case of an earth fault, and it is, therefore, possible to continue the power supply. Because of the low earth fault current, arcs are often self-extinguishing, and the grid can still be operated during a persisting earth fault. [1]

The major disadvantage is that the healthy phases experience a phase-earth voltage increase by the factor $\sqrt{3}$. The increased voltage puts additional stress on the isolation and can lead to further, subsequent earth faults in the healthy phases. It is recommended that an earth fault should not persist for more than two hours to keep the likelihood of a double earth fault as low as possible. [1]

2.2.3 Low Impedance Neutral Point

In networks with a low impedance neutral point, one or more neutral points of transformers are grounded via current limiting impedance R_E. In this case, every earth fault occurs as a short circuit with corresponding high short circuits currents. Each earth fault has to be switched off through the selective protective devices. The required maintenance level of the power supply is lower than in the previous neutral point systems. [3][6]

Low impedance networks are split into two types:

- ones in which the impedance of the neutral point is zero (solidly grounded) and
- ones in which the neutral point is grounded with an impedance greater than zero (low-impedance grounded neutral). [1]

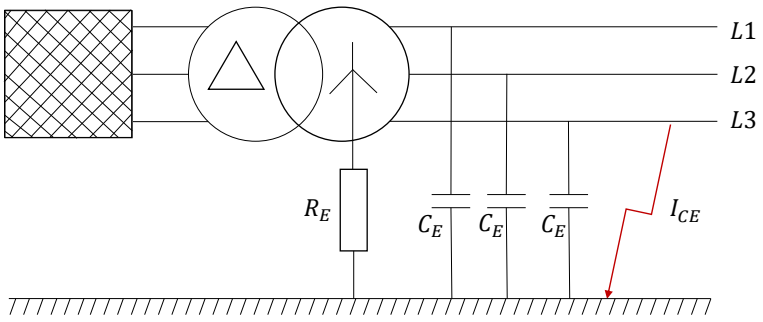


Figure 2-5, Low impedance grounded network [3]

The advantage of a low-impedance neutral system is that in the case of a single-pole earth fault, the voltages in the healthy phases experience a lower increase than in networks with insulated or resonant grounded neutral points.

$$I_{CE} = \frac{U_n}{\sqrt{3}R_E} \qquad 2\text{-}3$$

The high short-circuit currents, which lead to a voltage drop at the earth resistance, are problematic. A rapid shutdown (≤0.5 s) means that protection and earthing conditions can be maintained despite the high fault currents.

2.3 Earth Faults in Compensated Networks

Earth faults are single-pole faults that appear in three-phase power systems. The level of the earth fault current depends on the neutral treatment and on the parameters of the grid. These currents are particularly smaller than short-circuit currents and are commonly under nominal currents.

In case of an earth fault in a compensated network, an additional fault circuit opens over the arc-suppression coil, to the line-to-ground capacities, as shown in Figure 2-6. Hereby, the arc-suppression coil is dimensioned in a way that the capacitive earth fault current I_{CE} is more or less compensated through the inductive current I_L from the arc-suppression coil at the fault location. The remaining earth fault current is called residual earth fault current. The residual earth fault current consists only of an uncompensable reactive current component, a compensable active current component, and some harmonic components. Through this approach, it is possible to maintain the power supply during an earth fault. [7][8]

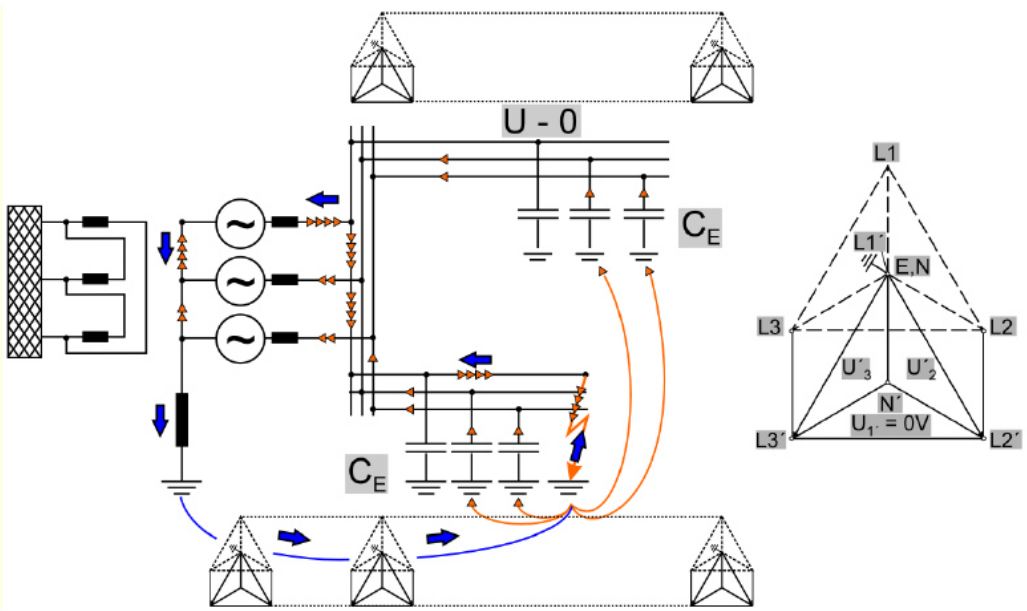


Figure 2-6, Earth fault in a compensated network [7]

The level of the residual earth fault results from the detuning of the arc-suppression coil. The detuning of the arc-suppression is defined as follows and describes the relationship between the capacitive earth fault current I_{CE} and the inductive current I_L caused by the arc-suppression coil. [9]

$$v = \frac{I_L - I_{CE}}{I_{CE}} \qquad 2\text{-}4$$

As the actual inductance L_D of the arc-suppression coil and the ideal inductance $L_{E,ideal}$ for full compensation. The capacitive earth fault current I_{CE} and the inductive current I_L are calculated as follows, as in [8]:

$$I_{CE} = \sqrt{3} \cdot U_n \cdot j\omega \cdot C_E \qquad 2\text{-}5$$

$$I_L = \frac{U_n}{\sqrt{3} \cdot j\omega \cdot L_D} \qquad 2\text{-}6$$

The detuning indicates whether the grid is operated overcompensated or undercompensated. In case of a positive detuning $v > 0$, the inductive current I_L is higher than the capacitive earth fault current. The grid is said to be overcompensated. If the detuning is negative $v < 0$, the grid is said to be undercompensated. [8]

The damping factor d is defined as the relation between the active power losses I_{act} of the arc-suppression coil and the capacitive earth fault current I_{CE}, as described in [9]. The damping factor is a measure for the ohmic component of the residual earth fault current.

$$d = \frac{I_{act}}{I_{CE}} \qquad 2\text{-}7$$

Usually, grids are operated overcompensated. The advantage of it is that in the case of line disconnections, the operating point is not going into the direction of the resonance point ($v = 0$) because the line-to-ground capacities are reduced by the disconnection. If the operating point is near the resonance point, a higher displacement voltage occurs (Figure 2-7), which might wrongly indicate an earth fault. [8][9]

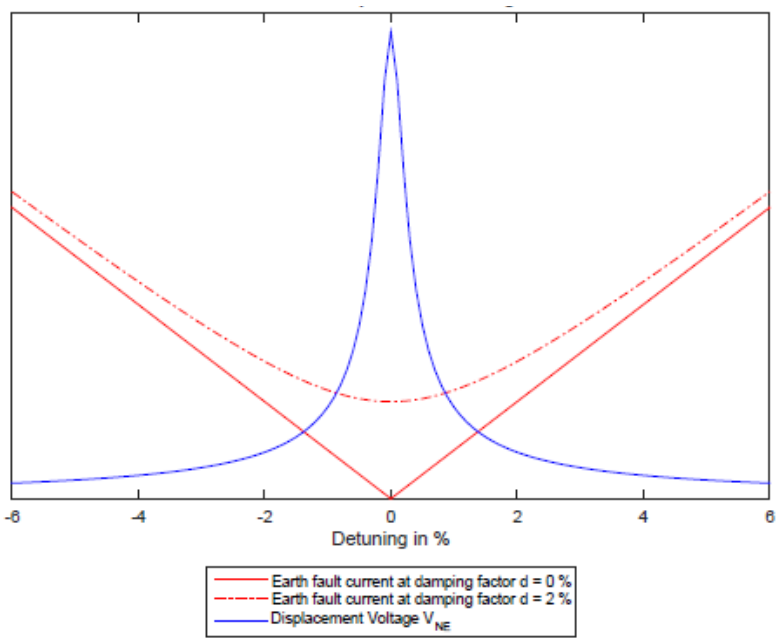


Figure 2-7, displacement voltage and residual earth fault current over detuning, [10]

2.4 Protection Philosophy

Protection technology is an essential element in the management of electrical power grids. It ensures the continuity of operations and the stability of the network by safely, quickly, and selectively switching off faulty network elements. Those faulty elements are switched on again as soon as possible once the

fault is rectified. A second requirement for the protection technology field is to avoid damage to humans, animals, and equipment through electrical touch voltages and arcs [6].

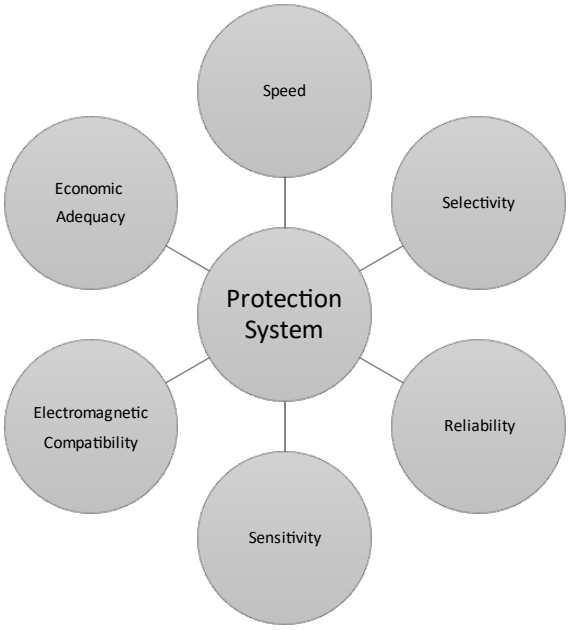


Figure 2-8, Requirements for protection devices in a protection system [4].

A modern protection system includes all the requirements shown in Figure 2-8. The protection devises from a protection system, which includes all components for measuring value processing, selective turnoffs, and all required information. A protection system has to have the highest availability. The availability is increased via the redundancy of equipment in the protection system [6].

A protection device must be able to reliably detect faults and ensure that faulty parts are switched off. To ensure security, the protection of a network is always carried out with redundant functions. This means that if a protective device fails, the protective function is taken over by upstream switching devices or fault detection devices working in parallel. Accordingly, a distinction is made between the main protection and reserve protection [11].

A protection system should only switch off the faulty area or the faulty components. This *selectivity* is achieved by the appropriate adaption of the tripping behavior of protection devices connected in a series. The main protection for short circuits needs to trip as fast as possible to avoid mechanical or thermal damage. A short error clarification time is needed. The total error clarification time consists of the detection period, the generation of the tripping order for the circuit breaker, and the time for the actual switching operation [11].

2.5 Earth-Fault-Detection

Earth-fault locating relays using classical methods, which are introduced later, refers to the identification of the earth-faulted branch. The whole outgoing branch is the object of protection. Classic earth-fault-locating methods are distinguished into transient and stationary methods. Transient methods have a high accuracy identifying earth-faulted branches, especially in cable networks, because of low-ohmic

transition impedances. Stationary methods use the displacement voltage and the zero-sequence current of outgoing branches to determine the earth-faulted branch. In this chapter, the following earth-fault-locating methods are introduced. [2]

- Transient method

Stationary methods:

- Wattmetric method
- Harmonic method
- Pulse method

2.6 Transient Method

The behavior of a single-pole earth fault can be described in three events. All events start at the same point but have a different period, as described in [12].

- Discharge of the earth-faulted line over the earth
- Charge of the healthy lines over the earth
- Stationary state of the earth fault

For the standard transient method, the charging process is used for the detection of earth-faulted branches. The direction (upstream, downstream) of a fault is determined through the signs of the zero-sequence current of a single branch, and the displacement voltage. If the displacement voltage reaches a certain threshold value for about 100ms, an earth fault is detected, and the signs of the displacement voltage are compared with the corresponding zero-sequence currents of the outgoing branches. If the signs differ from each other, the earth-faulted branch is found. If the signs are equal, the earth fault is located upstream. [12][2]

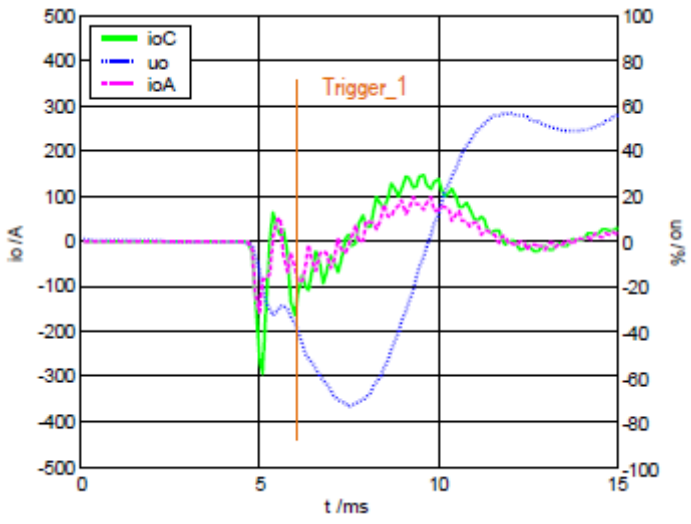


Figure 2-9, two healthy branches during a low ohmic earth fault, [12]

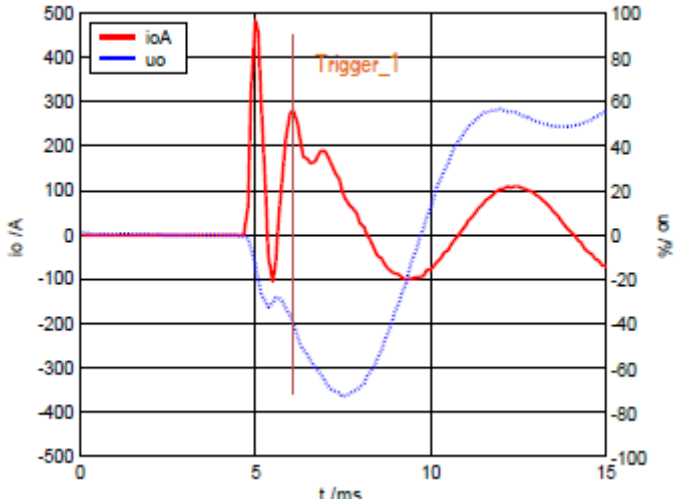


Figure 2-10, earth-faulted branch during a low ohmic earth fault, [12]

Figure 2-9 shows the displacement voltage and the zero-sequence currents of two healthy branches. The zero-sequence currents and the displacement voltage have the same sign. The threshold value for an earth fault was assumed at 40%. Figure 2-10 displays the earth-faulted branch, which is identified through different signs of the zero-sequence current and the displacement voltage. [2]

The standard transient method is only functioning when the transition impedance on the fault's position is lower than 50Ω. The following section describes the advantages and disadvantages as in [2].

Advantages:

- Not sensitive to angle deviations between the zero-sequence current and the displacement voltage
- The Holmgreen circuit can be applied to measure the zero-sequence current.

Disadvantages:

- Only suitable for earth faults with a transition impedance lower than 50Ω
- Only transient events can be analyzed in this method, and the method cannot be repeated.
- Sensitive to high circuit currents
- Analyzing such narrow time slots can lead to an incorrect decision on direction.
- It is difficult to determine and verify the threshold value for an earth fault.

The transient method can be improved by the qu-algorithm. The qu-algorithm takes advantage of the fact that the voltages of the healthy lines increase by factors of $\sqrt{3}$. The zero-sequence voltage of the branches can be described as follows [2]:

$$u_0(t) = u_0(t_0) + \frac{1}{C_{eqB}} \int_{t_0}^{t} i_{0B}(\tau)\, d\tau \qquad 2\text{-}8$$

This demonstrates that the zero-sequence voltage can only be built up when a current charges the line-to-ground capacities. The integral of the current describes the electrical charges, and the zero-sequence voltage is proportional to the electrical charges, assuming $u_0(t) = 0$. [2]

In Figure 2-11, the charge q is plotted on the y-axis and the zero-sequence voltage on the x-axis. The slope of the lines of the healthy branches B and C is the inverse value of the healthy branches' line-to-ground capacity.

Because of the reversed sign of the zero-sequence current of the earth-faulted branch, the model description is not working anymore. In compensated networks, the earth-faulted branch does not displace a line in the diagram, as shown in Figure 2-11. [2][12]

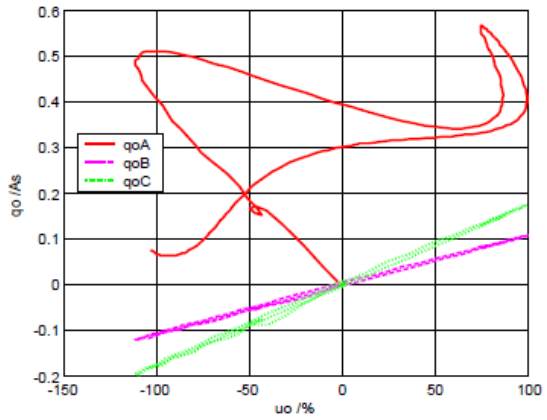


Figure 2-11, qu-diagram of a low ohmic earth fault in branch A, [2]

Through this approach, transient earth faults with a transition impedance of a few kΩ at the fault position can be detected. [2]

2.7 Wattmetric Method

In the case of an earth fault in a compensated network, the inductive current from the arc-suppression coil and the capacitive earth fault current has almost the same amount. The 50 Hz residual current consists of a capacitive component and an ohmic component, the so-called active residual current. In the wattmetric method, the amplitude and direction of the active residual current are determined. Through this, the earth-faulted branch can be identified. The applied $\cos(\varphi)$ circuit determines the active component through the measuring of the zero-sequence current and displacement voltage. The direction of the residual active current indicates whether the earth fault is located upstream or downstream. [2]

The active residual current is very low, and the result of this method is very sensitive to angle deviations of the current and voltage transformers. The active residual current depends on the impedances in the zero-sequence network, and especially from the arc-suppression coil. The method can be improved by

an increase of the active residual current by connecting a resistor in parallel to the arc-suppression coil, as shown in Figure 2-12. [2]

As in [2], there are several approaches to increase the active residual current:

- A permanent increase of 5 to 10A
- The residual active current increase of 5 to 20A during the healthy operating time to a few seconds after an earth fault occurs
- Short-term increasing in the area of 300A
- Short-term increasing in the area of 1200 to 2000A (KNOSPE).

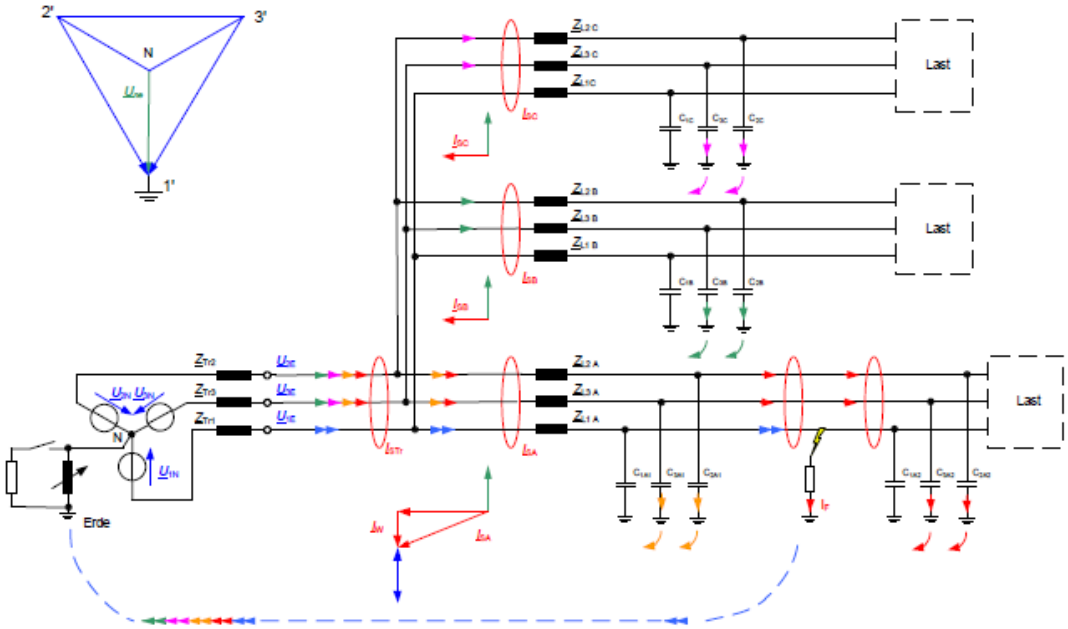


Figure 2-12, increasing of the active residual current, [2]

If the resistor is only switched on for the short-term, the detecting devices must have directional indicators with reset inputs or an automatic reset after a defined time. As in [2], the advantages of the wattmetric method with an increased active current are:

- Reduced requirements on the angle accuracy of the current and voltage transformer
- It can be used for switching faulty branches.

The disadvantages are:

- It is not functioning in ring topologies because of phase splitting
- Stationary earth fault is assumed
- Technical effort because of the complex circuit for the increasing of the active residual current
- Only short-term connection of the parallel resistor is possible because of the generated heat.
- With a higher current, the step and touch voltages must be considered.
- It is possible that with long lines, the required current at the fault position is not reached.
- Directional indicators with reset inputs or an automatic reset after a defined time are needed.

2.8 Harmonic Method

As same as the previous methods, the harmonics are used to detect earth-faulted outgoing branches of a substation. Through the harmonics, the compensated network can be approximated as an isolated network because the impedance of the arc-suppression coil is negligible compared to the very low impedances of the line-to-ground capacities.[2]

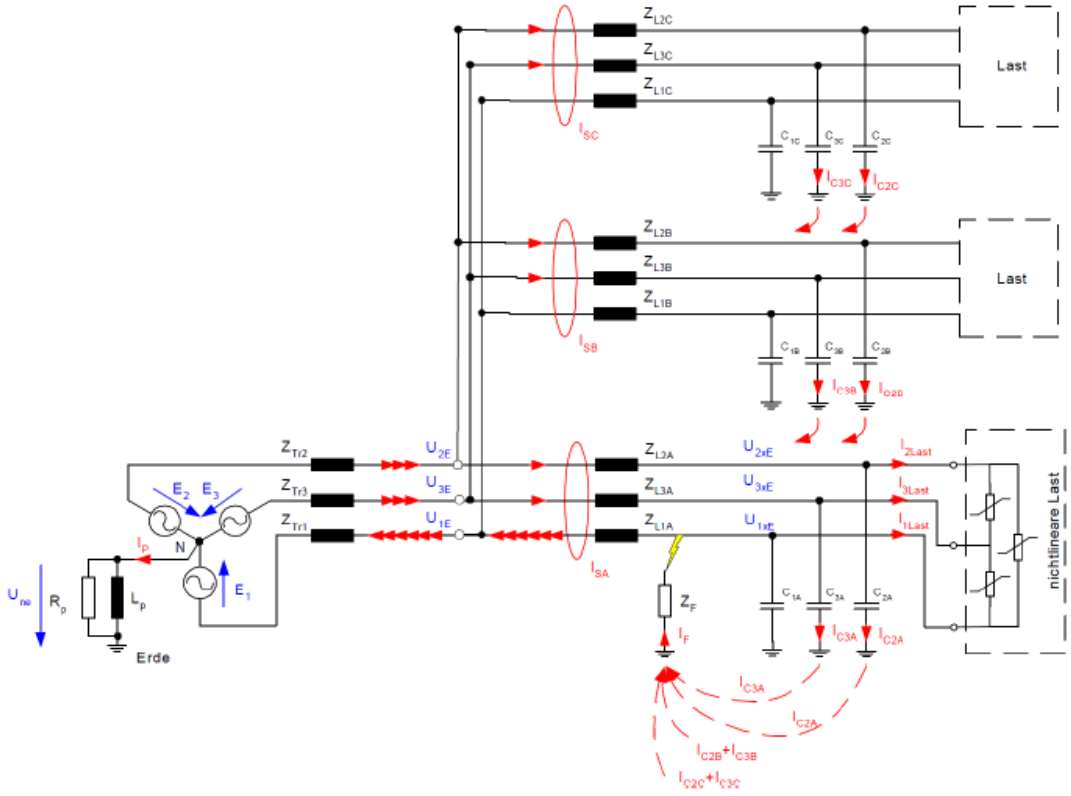


Figure 2-13, a compensated network for the harmonics, [13]

Harmonics occur at the earth-fault position. The sum of the harmonic currents at the affected earth-faulted outgoing branch is equal to the total sum of the harmonic currents at all healthy branches. The sum current of the healthy branches is capacitive and leads the displacement voltage by 90°. The sum current of the affected outgoing branch is inductive and lags the displacement voltage by 90°. [2][13]

These conditions can be used to detect an earth-faulted branch. One the one hand, the direction of the harmonic sum current in relation to the displacement voltage can be used. The distinction is made between a capacitive current (no earth fault), and an inductive current, which indicates an earth fault. On the other hand, all harmonic sum currents of the branches can be compared. The branch with the relatively highest current indicates an earth fault. [2][13] [14]

By applying a reactive power directional relay and an upstream 250 Hz filter, earth faults can be detected for specific branches. In pure cable and pure overhead line networks, the harmonic method operates reliably. Unreliability comes up when the method is applied to mixed networks. In branches that begin with overhead lines in combination with following cable sections, the resonant point of the series

resonant circuit is overstepped very fast, which results in a current reversal at the fault position. [2][13] [14]

Figure 2-14 shows the serial resonant point of the harmonic in a 20kV medium voltage overhead line network. The length of the overhead line is shown on the x-axis, the capacitive zero-sequence current on the y-axis. As long as the network is operating on the left side, the harmonic method is functioning. [2][13][14]

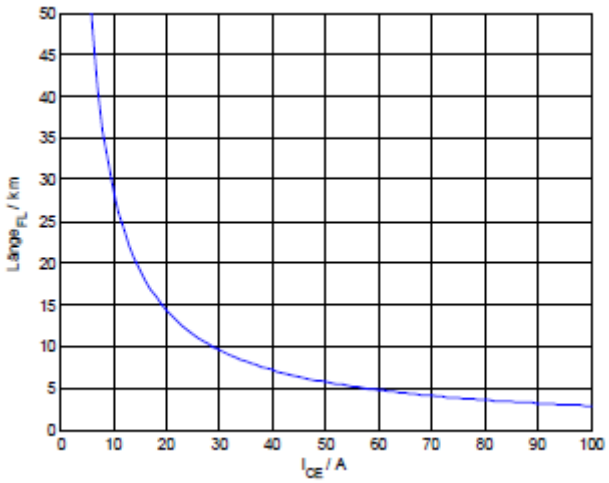


Figure 2-14, resonant frequency 250 Hz, [2]

Advantages of the harmonic method (as in [13]):

- The compensated network can be considered as an isolated network.
- Easy realizable and robust
- Not sensitive to angle errors between the displacement voltage and the zero-sequence-current

Disadvantages of the harmonic method:

- Stationary state of the earth fault is necessary
- Not compatible in ring topologies
- A healthy residual network is necessary
- Through a series resonant circuit in mixed networks (overhead lines and cables), a more detailed verification of the configuration is needed.
- Harmonic components of the phase-to-phase voltage must be present.
- The Holmgreen circuits need to be tuned exactly.

2.9 Pulse Method

The pulse method is used to determine earth-faulted branches by cyclical parallel connections of a capacitor to the arc-suppression coil. By this, the reactive current at the fault position changes. These pulses can only be measured from the clock generator to the fault position, as shown in Figure 2-15. The earth-faulted branch can be detected by the zero-sequence current modulation. The required

current change is usually in the area of 2 to 3% of the capacitive earth-fault current. In addition, the operation of the network must be overcompensated in order to be able to clearly record the pulses along the lines to the point of failure. [13]

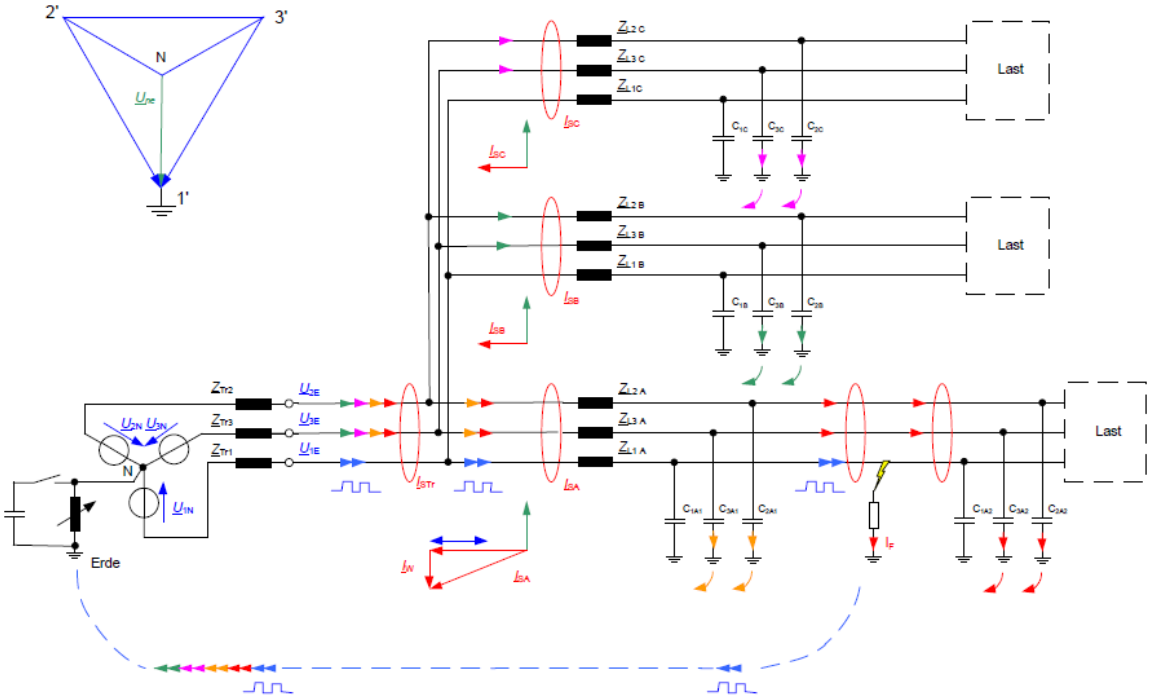


Figure 2-15, pulse method, [2]

The pulse method functions below transition impedances of 300Ω. Five out of seven pulses must be detected correctly to indicate earth faults with high accuracy. During the pulse time, the zero-sequence network should not be changed. The first results of the pulse method can be expected after around 25 seconds. [2]

Figure 2-16 shows the principle of the pulse detection method. The pulses coming from the clock generator to the fault position are identifiable by pulse locating relays. The earth fault is located in the segment between the last detected pulse and the first undetected pulse. The more pulse locating relays that are implemented, the more precise the narrowing. [2]

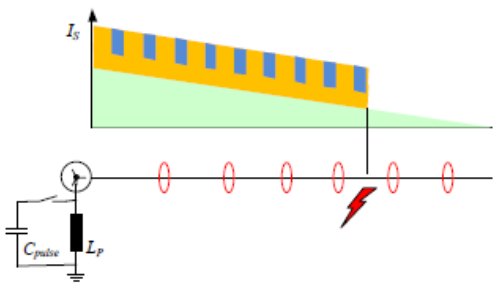


Figure 2-16, principle pulse detection [2]

If a higher amount of pulse locating relays is installed, it is essential to operate the grid overcompensated at least as high as the pulse currents are. Otherwise, a wrong segment would be detected because of the elimination of the pulse signals. [2]

In ring topologies, the pulse method does not work. The pulse current flows from both directions to the position of the earth fault. If the amount of the pulse current is high enough, an earth fault is indicated along the whole ring, and it is not possible to distinguish between the single segments. If a fault is located in the middle of the ring, the pulse currents are cut in half. [2]

As in [2], the advantages and disadvantages of the pulse method are as follows.

Advantages:

- Since only the current is evaluated, the pulse method is not sensitive to angle errors between the displacement voltage and the zero-sequence current
- Holmgreen circuit can be applied
- More detailed locating through the measuring of the zero-sequence current is realizable

Disadvantages:

- Stationary earth fault for 30 seconds is assumed
- An overcompensated grid is assumed, which can be in conflict with other conditions, like the limits of the touch and step voltages
- Detailed locating via locating pulse relays is not possible in ring topologies
- Sensitive to harmonics in the sum current
- Symmetrical pulsing is only suitable for low fault resistance. Unsymmetrical pulsing usually works for a fault resistance below 300Ω
- Does not work for repeating and intermittent faults
- The amount of the required overcompensation depends on the active component of the grid
- Extinguishing power of decentralized coils has to be considered
- It does not work for undercompensated grids. It creates a “blind section” in the line, which can be interpreted as fault position

What all of the above-introduced methods have in common is that only the earth-faulted branch is detected, rather than an earth-faulted section along a branch. The pulse method is able to differ in sections, but only if pulse relays are installed in locale grid stations. This major disadvantage leads to longer troubleshooting times, which further result in a higher isolation stress for the healthy components.

Neural networks are used to try to outweigh this disadvantage. It was investigated how neural networks can be applied to distinguish in earth-faulted sections along a branch by only measuring the voltages and currents of the earth-faulted branch in the substation.

3 Modelling and Simulation

3.1 Introduction

All neural networks were trained on before simulated data, generated through the software DIgSilent PowerFactory. With this approach, all data samples were already labelled automatically by DIgSilent PowerFactory. The labelled samples were used later to train neural networks.

The simulations for the data generating process differ regarding the tasks. For the fault classification task, a reduced exemplary grid model was used to simulate different kinds of faults in order to determine whether a neural network can be used to classify the faults. For the earth-fault-locating task, different kinds of grid topologies, which differed in complexity, were simulated to investigate the limits of earth-fault-locating neural networks.

3.2 DIgSILENT PowerFactory

DIgSILENT PowerFactory is a widely used power engineering software. The software was used for EMT (Electro-Magnetic-Transients) fault simulations in compensated medium voltage networks. DIgSILENT PowerFactory has the advantage that simulations can be executed, while variating grid parameters by an external python script. With this, simulations with a wide range of different fault possibilities could be completed almost automatically.

Most of the network calculating programs use two types of dynamic simulation methods, EMT (electromagnetic transients) and RMS (root mean square). In EMT simulations, voltages and currents are displayed as instantaneous values. EMT considers very high-frequency phenomena, such as dynamic behaviours of passive components. It is used for simulations of electromagnetic dynamic balancing processes. In contrast, in RMS simulations, transients are neglected, and only the fundamental frequency of the root mean square values are calculated. Compared to EMT, the advantage of RMS simulations is in the shorter required simulation times, which allows for simulating complex grid systems in a short time. [15][16] Due to the fact that transient processes are also considered, the EMT simulation method is used for the further simulations.

The simulation results, the measured voltages, and currents in the substation of the simulations were stored as CSV files (comma-separated values), including the fault position, type of fault, and all parameters which varied during simulations. By the storage in CSV files, the data was easily available for the data preparation for the neural networks.

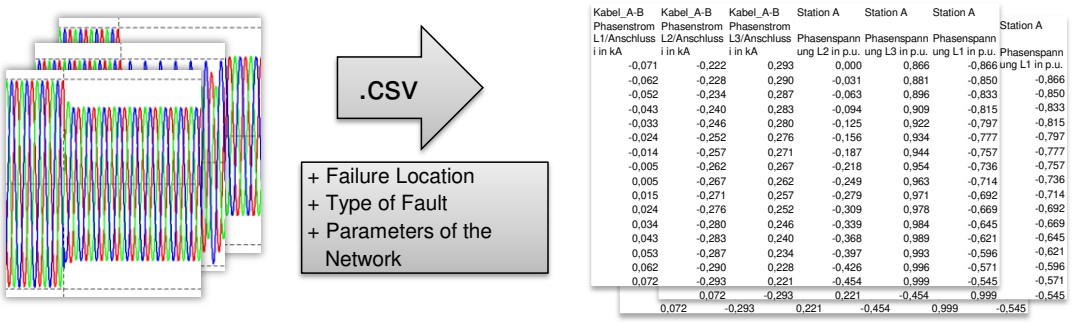


Figure 3-1, simulated faults were stored as CSV files

3.3 Simulation Model

The grids were created in the software DIgSILENT PowerFactory. The simulated and generated data served as a data basis for the development of neural networks. The following grid parameters were selected for the fault classification task as well as for the earth-fault-locating task.

Grid and transformer	Unit	Reference Level
Nominal voltage	kV	20
Grid, short circuit level	MVA	3000
Transformer power	MVA	40
Transformer short circuit voltage		15%

Table 3-1, grid parameters

In all grids, the same cable is used.

Cable	Voltage level	R^1	X^1	C^1	C^0	R^0	X^0
	kV	$\frac{\Omega}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$	$\frac{\mu F}{\mathrm{km}}$	$\frac{\mu F}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$
NA2XSY 3x240rm	12/20	0.1281	0.0974	0.31	0.3216	0.5125	0.3896

Table 3-2, cable parameters [15]

3.3.1 Sensitivity Analyses

Sensitivity analyses are used to evaluate how sensitive a mathematical model is through minor modifications of input parameters. Particularly in the field of economic sciences, sensitivity analyses find applications in investment and project management to assess the impact of long-term variables or unknown costs. The analysis determines which single components are the biggest risk for a project or an investment over time.

The sensitivity analysis was used to estimate the influence of variables that may have an impact on the result of short circuits along the cable. Below, in Figure 3-2, a 3-phase short circuit at the end of Cable1 is shown. The medium voltage cable network consists of an external grid, a transformer, two cables, and two loads at the end of it.

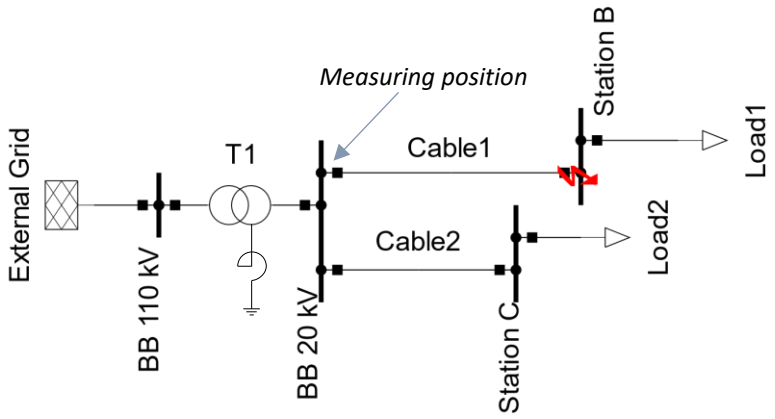


Figure 3-2, Medium voltage cable network with a short circuit in Cable1

In the following, the 3-phase short circuit was calculated and applied to a sensitivity analysis. Figure 3-3 shows the equivalent circuit diagram of the medium voltage network, including the 3-phase short circuit (red arrow).

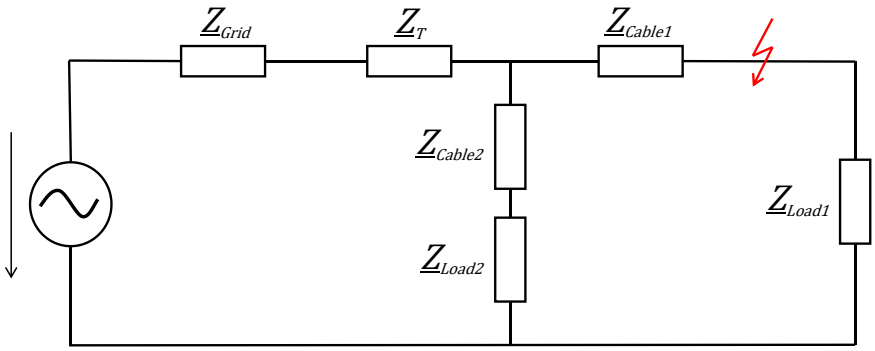


Figure 3-3: Equivalent circuit diagram including a 3-phase short circuit

In the next steps, the single impedances were calculated. The sensitivity analysis must determine a reference short circuit value of the whole network that serves as a benchmark for the modifying input variables. First, the grid impedance was determined by the nominal voltage squared multiplied by the voltage factor of 1.1 divided by the grid short-circuit level.

$$\underline{Z}_{Grid} = \frac{c \cdot {U_N}^2}{SCL} = \frac{1.1 \cdot 20\,kV^{\,2}}{3000\,MVA} = 0.1467\,\Omega \qquad 3\text{-}1$$

The transformer impedance is given by the short-circuit voltage uk in percent times the nominal voltage squared divided by the apparent transformer power.

$$\underline{Z}_{Tr} = \frac{uk \cdot {U_N}^2}{S_T} = \frac{15\% \cdot 20\ kV^{\,2}}{40\ MVA} = 0.15\ \Omega \qquad 3\text{-}2$$

The impedance of the $Load2$ was determined by nominal voltage squared divided by the load power plus the tangent of an angle.

$$\underline{Z}_{Load2} = \frac{U^2}{S} = \frac{20\ kV^2}{16\ MVA} = 25\ \Omega \qquad 3\text{-}3$$

Both line impedances are given by the cable length of $6\,km$ times the resistance and the inductive reactance as follows:

$$\underline{Z}_{Cable1} = \underline{Z}_{Cable2} = l \cdot \sqrt{R'^2 + j\,{X'^2}_L} = 6\ km \cdot \sqrt{0.1281^2\ \frac{\Omega}{km} + 0.0974\ ^2 j\,\frac{\Omega}{km}} = 0.9654\ \Omega \qquad 3\text{-}4$$

The calculation of the 3-phase short circuit current the substitute source method was used. Since only the current amplitude at the short circuit location is of interest, it was only calculated with absolute values. In the following, the equivalent short circuit diagram with the substitute source method is shown.

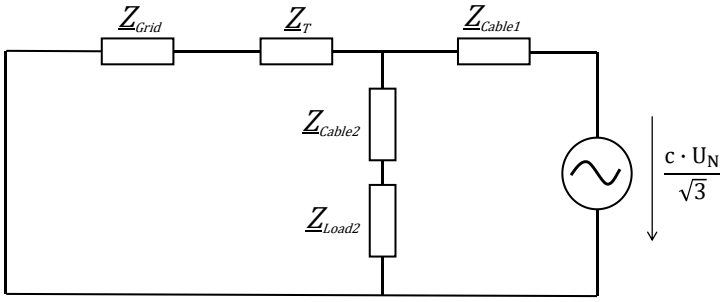


Figure 3-4, Equivalent short circuit diagram with the substitute source method

Out of this diagram, the absolute short circuit current was calculated as follows. This value was used as a reference value for the sensitivity analyses.

$$I''_k = \frac{c \cdot U_N}{\sqrt{3} \cdot \underline{Z}_{total}} = \frac{1.1 \cdot 20\ kV}{\sqrt{3} \cdot 2.5139\ \Omega} = 5.0525\ kA \qquad 3\text{-}5$$

Through the sensitivity analyses, the single input variables were varied between 10% and 20% depending on their possible range. The following Table 3-3 shows the reference level and the range of variation of each input variable.

Variable	Unit	Reference Level	Range of Variation
Nominal voltage	kV	20	
Grid, short circuit level	MVA	3000	
Transformer power	MVA	40	
Transformer short circuit voltage	V	15%	
Power Load2	MVA	16	± 20%
Power Factor Load2		0.9 ind.	± 10%
Length Cable1	km	6	± 20%
Impedance Cabel1	$\frac{\Omega}{km}$	0.1281 Ω +0.09739 $j\Omega$	

Length Cable2	km	6	± 20%
Impedance Cabel2	$\frac{\Omega}{\text{km}}$	$0.1281\,\Omega$ $+0.09739\,j\Omega$	

Table 3-3: Sensitivity Analyses, Reference Level, Variables and Range of Variation

Figure 3-5 shows the absolute result in percent depending on the single variation of each input variable. The slope indicates the change. The higher the slope, the more significant is the effect on the result. As shown, the length of $Cable1$ has by far the biggest impact on the result. The influence of $Cable2$ stays almost the same throughout the entire analysis, whereas the power factor and the power of $Load2$ have a bigger impact but still not as much as the cable length.

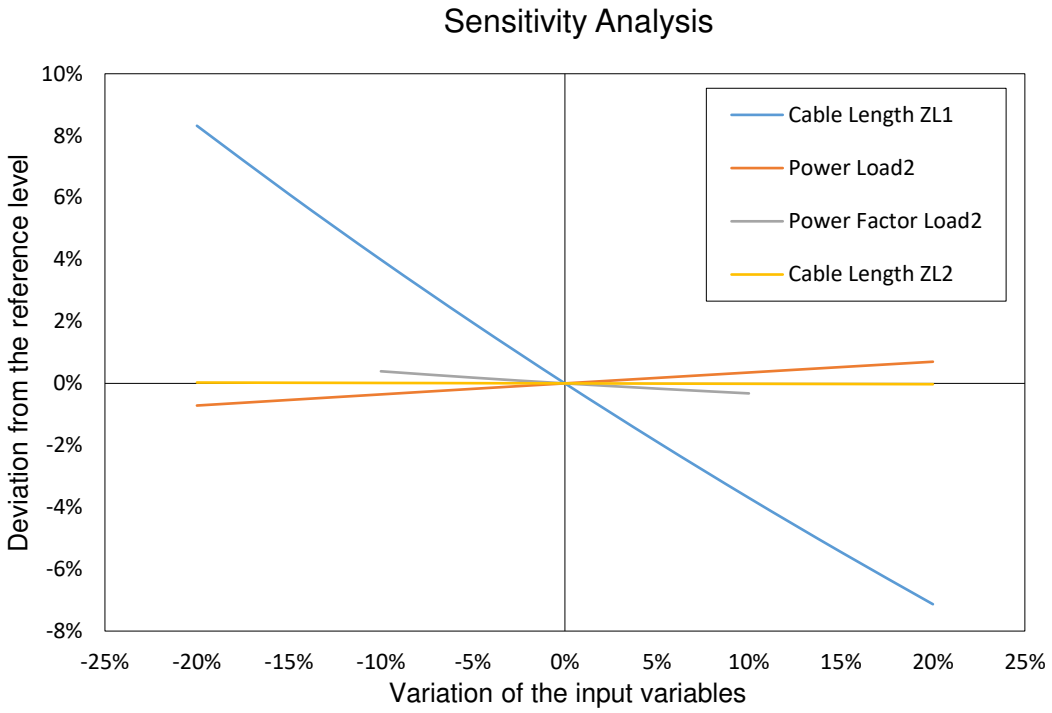


Figure 3-5: Outcome of the Sensitivity Analyses

As mentioned before, the indicators of the sensitivity analysis are the single slopes from the input variables in the diagram. The greater the slope of a single variable, the higher is the effect on the result and vice versa. For the impact, it does not matter whether the slopes are positive or negative.

As the diagram above shows, the longer the cable length of $Cable1$ and, therefore, the distance from the feeding busbar, the higher is the influence of the short circuit current. In contrast, the short circuit current of $Load2$ and its power factor has a very flat slope. Thus, the impacts on the result are low.

Based on the outcome of the sensitivity analyses, the main focus in the data generation via the simulation will rely on the cable length and on the short circuit position along the affected cables. In sum, this sensitivity analysis allowed assessing the effects and sources of uncertainties in the interest of generating suitable data.

3.3.2 Reduced Model for Fault Classification

For the fault classification task, the topology of the grid model applied in the sensitivity analysis was used. The exemplary 20kV reduced compensated medium voltage grid is shown in Figure 5-1. It was used for data generation of different kinds of faults. Along cable1, faults were simulated. The generated data was later used to train, validate, and test a neural network to classify these faults. The results of the neural network are shown in chapter 0.

Simulated Faults:

- 3-phase short circuit
- 2-phase short circuit L1–L2
- 2-phase short circuit L2–L3
- 2-phase short circuit L1–L3
- Earth fault L1
- Earth fault L2
- Earth fault L3

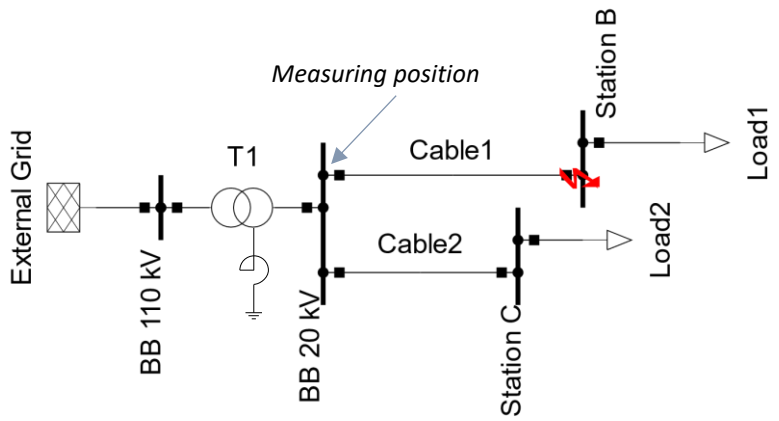


Figure 3-6, a grid for fault simulation

Based on the results of the sensitivity analysis, the grid parameters were varied during the simulation, for each simulated fault. The focus in the simulations was on the cable length and the fault location along the affected cable.

	Variated	Steps
Length of cable1	From 0.5 to 6 km	0.5 km
Fault position	From 10% to 100%	10%
Load1	From 2 to 4 MW	2 MW
Load2	From 1 to 5 MW	4 MW

Table 3-4, varied parameter during simulation

The length of cable1 was varied from 0.5 to 6 km in 0.5 km steps. The fault position relative to the length of cable1 ranged from 10 to 100% in 10% steps. The power of load1 of the faulty branch varied from 2

to 4 MW in 2 MW steps. The power of the load2 of the healthy residual grid varied between 1 and 5 MW in 4 MW steps.

The currents and voltages were measured in the substation at the faulty outgoing branch, as is usual in operating systems. With this, the currents and voltage curves of the simulation were saved and stored as CSV files, including the type of fault, the failure location, and all the parameters, which changed during the simulation. Each type of fault was simulated on every possible variation of the parameters. Thus, the fault simulation ran through all possible variations and resulted in 3,360 data samples. The data served as a database for the machine learning algorithm, especially for neural networks. The results are presented in chapter 0.

3.3.3 Verification of the Simulation Model for Earth Fault Simulation

Extended grid models were used in the earth-fault-locating task to examine the strength and weakness of earth-fault-locating neural networks. First, an exemplary reduced grid model was applied to verify DIgSILENT PowerFactory's results, shown in Figure 3-7. The same grid topology was later used to generate data. For the verification, the detuning v of the inductance current of the arc-suppression coil and the capacitive earth fault current is zero. Thus, the grid runs fully compensated. The grid and transformer parameters were selected, as shown in Table 3-5. Each of the seven outgoing branches consists of a 10km medium voltage cable. The cable parameters are shown in Table 3-6.

Grid and transformer	Unit	Reference Level
Nominal voltage	kV	20
Grid, short circuit level	MVA	3000
Transformer power	MVA	40
Transformer short circuit voltage		15%
Detuning v of the grid		0%
Total cable length	km	50

Table 3-5, grid parameters

Cable	Voltage level	R^1	X^1	C^1	C^0	R^0	X^0
	kV	$\frac{\Omega}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$	$\frac{\mu F}{\mathrm{km}}$	$\frac{\mu F}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$	$\frac{\Omega}{\mathrm{km}}$
NA2XSY 3x240rm	12/20	0.1281	0.0974	0.31	0.3216	0.5125	0.3896

Table 3-6, cable parameters [15]

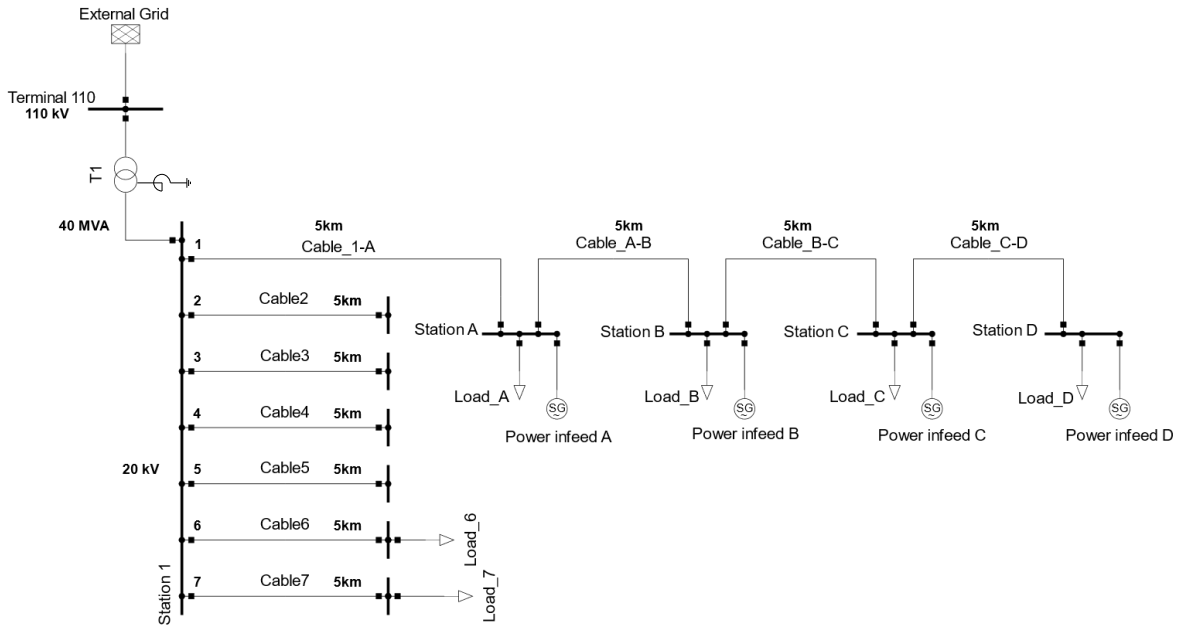


Figure 3-7, an exemplary reduced grid for earth fault simulation

First, the capacitive earth fault current I_{CE} was calculated, as well as the needed reactive impedance X_P and the current I_P of the arc-suppression coil. Next, the value of the reactive impedance X_P was added to the DIgSILENT PowerFactory simulation to compare the results of the simulation to the results calculated by hand.

The total earth capacitance of the network is the product of the line-to-ground capacitance per km and the total cable lengths in the network.

$$C_E = C^0 \cdot total\ cable\ length = 0.3216\ \frac{\mu F}{km} \cdot 50\ km = 16.08\ \mu F \qquad 3\text{-}6$$

With this, the capacitive earth fault current I_{CE}, which depends on the nominal frequency and on the line-to-ground capacitance, is as follows:

$$I_{CE} = \sqrt{3} \cdot \omega \cdot U_N \cdot C_E = \sqrt{3} \cdot \omega \cdot 20\ kV \cdot 16.08\ \mu F = 174.9954\ A \qquad 3\text{-}7$$

The reactive impedance X_P and the inductance L_P of the arc-suppression coil, to fully compensate the capacitive earth fault current I_{CE}, was determined as below:

$$X_P = \frac{1}{3 \cdot \omega \cdot C_E} = \frac{1}{3 \cdot \omega \cdot 16.08\ \mu F} = 65.9846\ \Omega \qquad 3\text{-}8$$

$$L_P = \frac{X_P}{\omega} = \frac{65.9846\ \Omega}{\omega} = 0.210036\ H \qquad 3\text{-}9$$

The inductive current I_P over the arc-suppression coil is the result of the nominal voltage divided by the inductance of the arc-suppression coil and the frequency:

$$I_P = \frac{U_N}{\sqrt{3} \cdot \omega \cdot L_P} = \frac{20\ kV}{\sqrt{3} \cdot \omega \cdot 0.150025\ H} = 174.9950\ A \qquad 3\text{-}10$$

Due to rounding errors, there is a slight difference to the capacitive earth fault current I_{CE}.

With the set value X_P of the reactive impedance of the arc-suppression coil, the software DIgSILENT PowerFactory calculated the capacitive earth fault current and the inductive current for the arc-

suppression coil. The results are shown in Table 3-7. In comparison, the results do not deviate from the calculation by hand.

Capacitive earth fault current I_{CE}	174.995 A
The inductive current of the arc-suppression coil I_P	174.995 A

Table 3-7, I_{CE} and I_P calculated by DIgSILENT PowerFactory

Next, it was examined how the three phase-to-earth voltages at station A, the displacement voltage, and the current react when it comes to an earth fault in the voltage maximum of phase L2 at the end of branch 1 of the grid in Figure 3-7. The results were compared with the theory in [3] and [2].

Figure 3-8 shows the measured phase-to-earth voltages. Since the voltages were measured in the substation, the earth-faulted phase L2 is overlaid with high-frequency voltages due to reflections of the traveling wave at each end of the network. The traveling wave originated at the earth fault position and spread in both directions. It is different when the voltages are directly measured at the earth fault position, as shown in Figure 3-9.

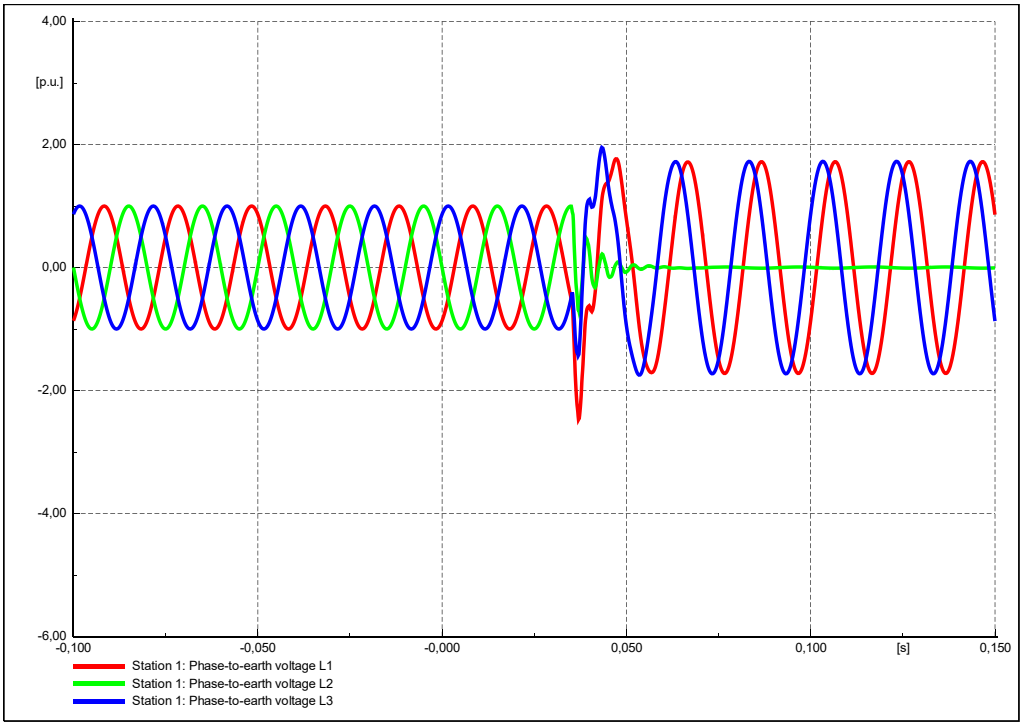


Figure 3-8, earth fault in a voltage maximum measured in the substation

By an ignition in the voltage maximum of the phase-to-earth voltage, the earth-faulted phase discharges and the network immediately tries to charge the healthy phases by the factor $\sqrt{3}$. This strong charging process leads to voltage peaks in the healthy phases before reaching the stationary state, which can be seen in Figure 3-8 and in Figure 3-9.

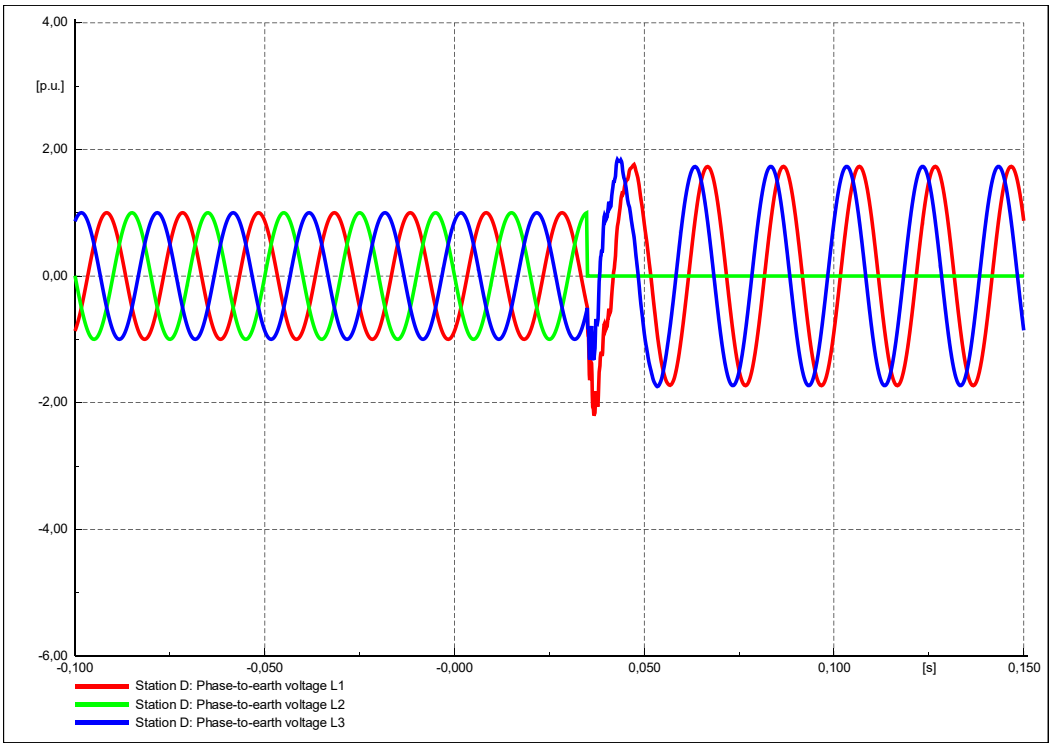


Figure 3-9, earth fault in a voltage maximum measured at the fault position

Figure 3-10 shows the displacement voltage and current over the arc-suppression coil. The current indicates that the earth fault current at the fault position is compensated immediately. Regarding the ignitions of earth faults in the voltage maximum, the displacement voltage does not include any DC components. The harmonics occur because of the transient response of the charging process in the healthy phases.

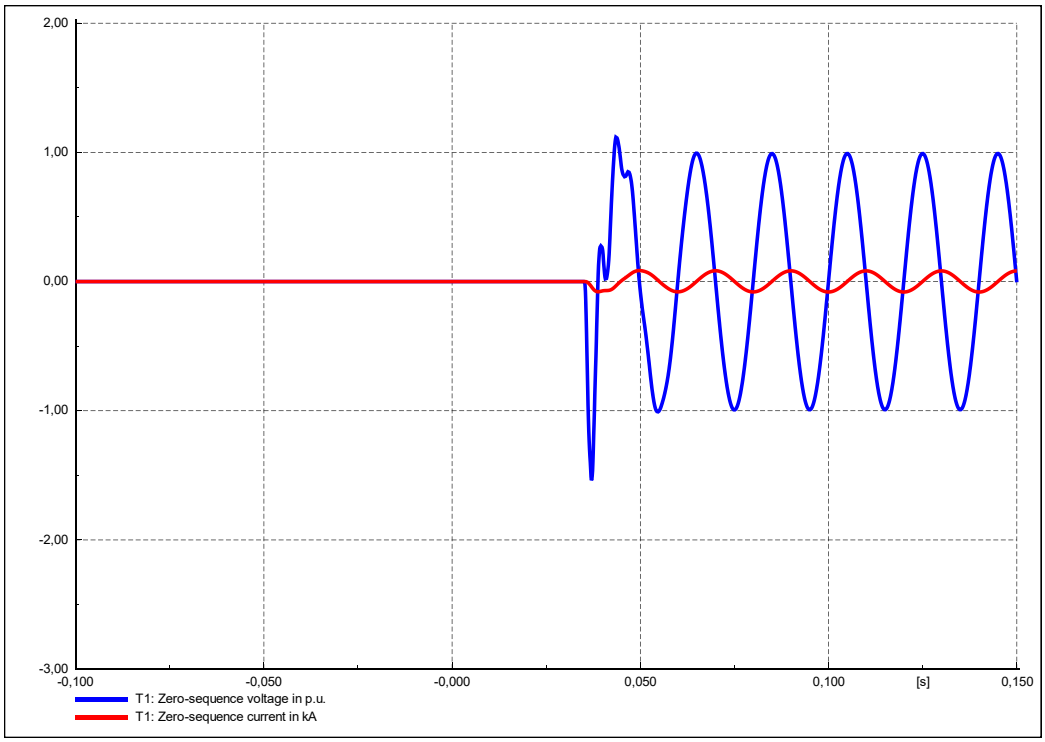


Figure 3-10, displacement voltage and current through the arc-suppression coil

In the following figures, it is shown how the network responds if the ignition of the earth fault is in the voltage minimum. As shown in Figure 3-11, the current over the arc-suppression coil contains a declining DC component, as well as the voltage of the earth-faulted phase L2 Figure 3-13. The value of the DC component depends only on the parameters of the arc-suppression coil. Different than before, with the ignition in the maximum, in the minimum, the discharging process is not applicable. Only the charging process of the healthy phases occurs.

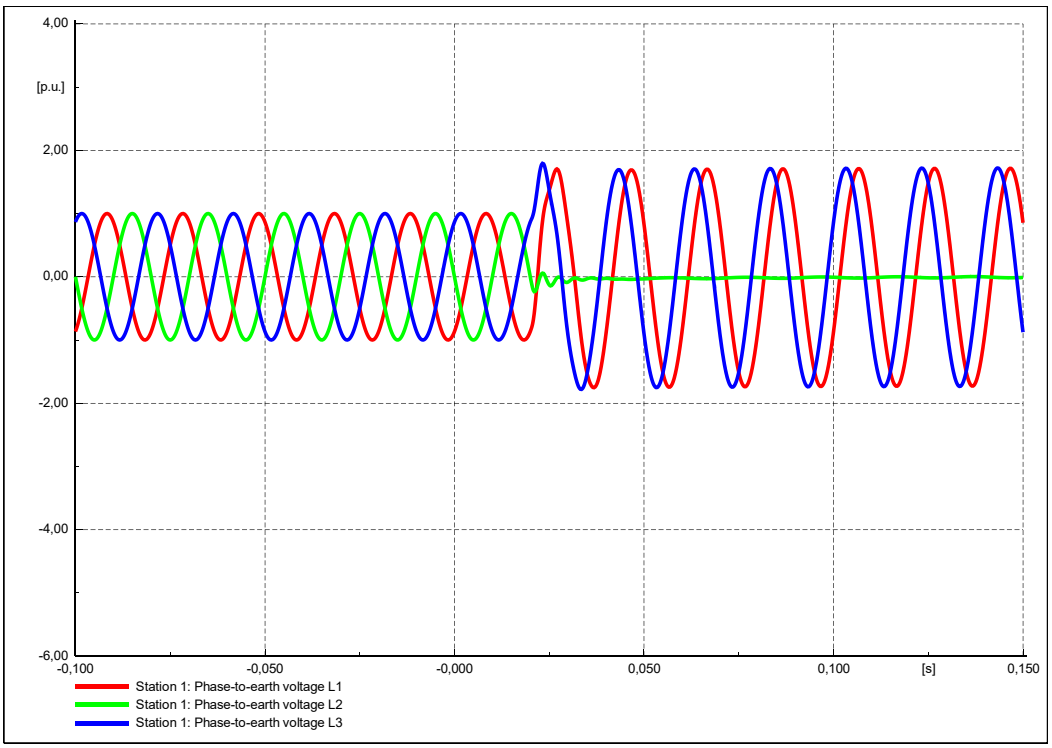


Figure 3-11, earth fault in a voltage minimum measured in the substation

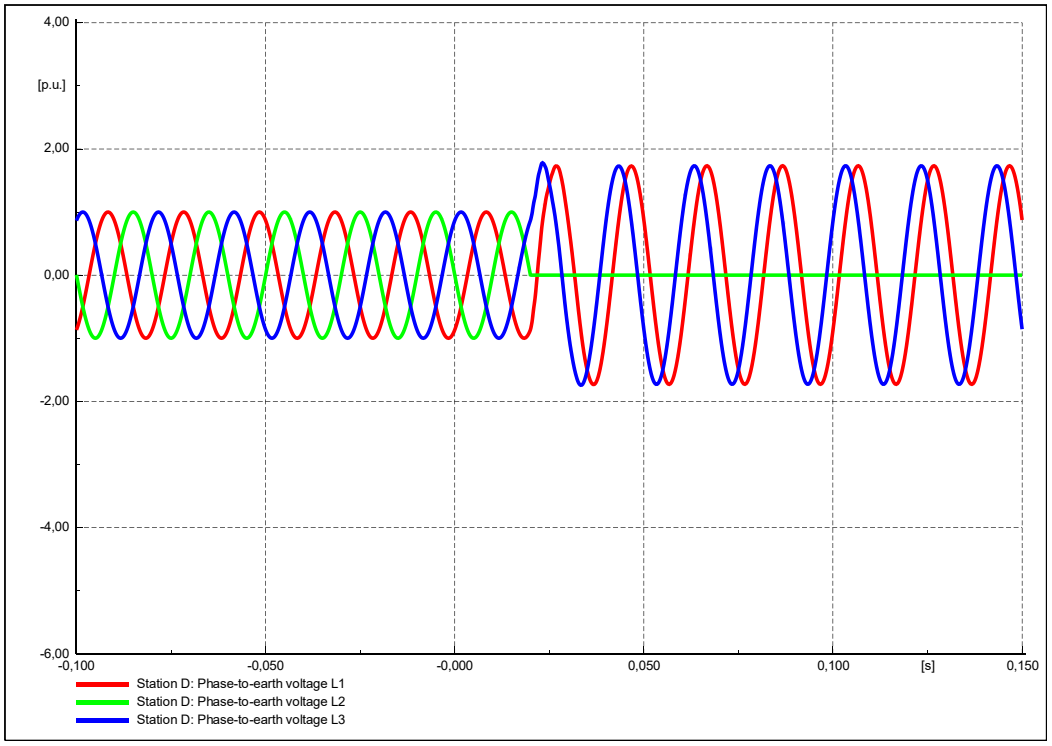


Figure 3-12, earth fault in a voltage minimum measured at the fault position

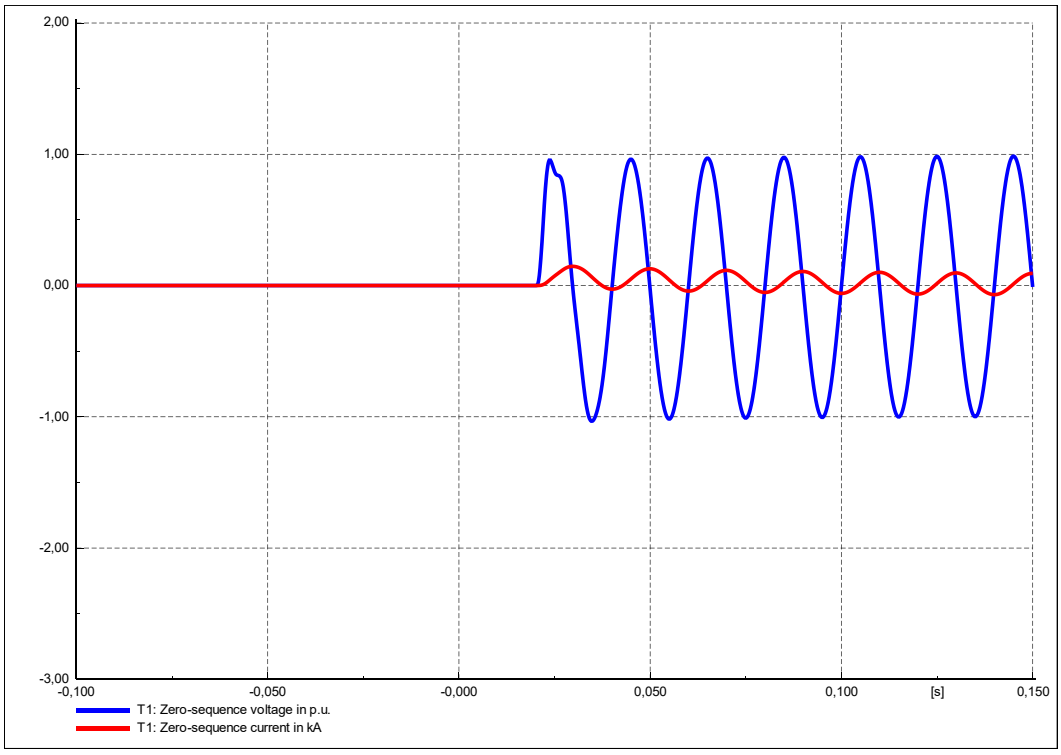


Figure 3-13, displacement voltage and current through the arc-suppression coil

The results of these measurements correspond with the theory in [3] and [2], and therefore indicate the correctness of the simulations. Since the first ignition of an earth fault usually occurs in or near the maximum of the phase-to-earth voltage, all earth fault ignitions were simulated in the phase-to-earth voltage maximum. [2]

3.3.4 Extended Grid Models for Earth Fault Localization

For the earth-fault-locating task, different grid models were simulated to investigate the limits of such earth-fault-locating neural networks. These extended grid models are introduced below. The method for data generation for the earth-fault-locating task was almost the same as for the fault classification task. The variations of the single grid parameters during the simulations were held on the outcomes of the sensitivity analysis. Different exemplary grid models were examined, which differed in complexity. All grid models were 20kV compensated medium voltage cable networks, that differed in cable lengths, number of local grid stations, and fanouts. The measuring position of the three voltages and currents was in the substation. The generated data was used to train neural networks for earth fault locating. The results of the earth-fault-locating neural networks are shown in chapter 5.2.

3.3.4.1 Model for Earth Fault Localization, 4 Stations

The first investigated exemplary grid model is shown in Figure 3-14. The 20kV compensated medium voltage cable network consists of four local grid stations. All such stations are connected with a 5 km cable. Each station consists of a power infeed and a load. Earth faults were simulated along each cable at branch 1 in 10% steps from 10% to 100% relative to the cable length. The aim of the earth-fault-locating neural network was to localize these simulated earth faults between the stations by classification.

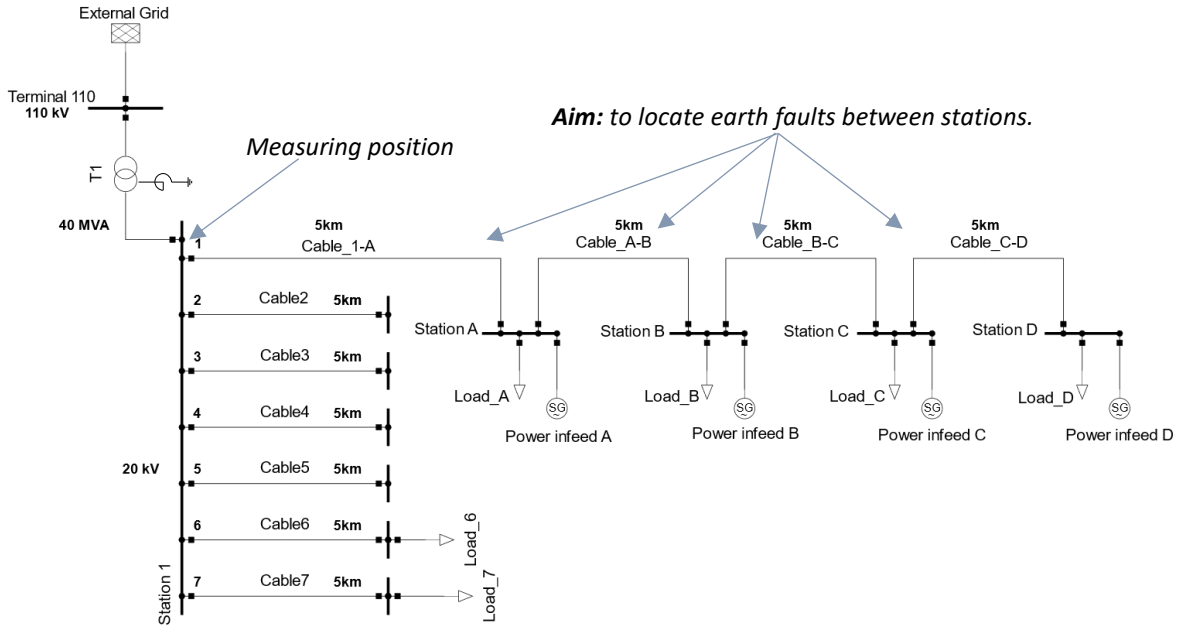


Figure 3-14, earth fault simulation with four stations on the affected feeder.

In Figure 3-14, the healthy residual grid consists of six branches, each with a 5 km cable line. The power of the residual grid is represented in branches 6 and 7. For simplification and their low impacts on earth faults in branch 1, branch 2 to 5 are without any loads. The grid operates 5% overcompensated.

As mentioned, the aim was to locate earth faults between the four stations through the classification by a neural network, whereas the voltages and currents measured in the substation were the input of the neural network. Earth faults were only simulated in phase L2 since the correct fault differentiation

through a neural network was part of the previous fault classification task. During the simulation, the following parameters were varied:

	Varied	Steps
Earth fault position	From 10% to 100%	10%
Loads at stations A, B, C and D	From 0 to 0.3 MW	0.3 MW
Load at branch 6	From 1 to 5 MW	4 MW

Table 3-8, varied parameter during simulation

The earth fault position relative to the cable's length ranged from 10% to 100% in 10% steps for each 5 km cable. The power of the four loads of the earth-faulted branch 1 ranged between 0 and 0.3 MW in 0.3 MW steps. The power of the load of the residual healthy branch 6 ranged between 1 and 5 MW in 4 MW steps.

The power infeeds at the four stations (Station A, Station B, Station C, and Station D) were constant at 0.15 MW. Thus, during simulation, if the power of a load in one station is 0 MW, the station fed in 0.15 MW, otherwise, the station draws power from the branch. The load at branch 7 was constant at 5 MW and did not vary. All loads and external branches operated with a power factor of 0.95 inductive.

	Constant
Power infeeds at station A, B, C, and D	0.15 MW
Load at feeder 7	5 MW

Table 3-9, constant parameter

Through this variation process, 1,280 data samples were generated. As before, the generated data samples were stored as CSV files and included the type of fault, the failure location, and all the parameters, which changed during the simulation.

3.3.4.2 Model for Earth Fault Localization, 9 Stations

The second exemplary grid model in Figure 3-15 consists of nine local grid stations and cable connections of different lengths, from 1 to 4 km. As in the previous grid model, the measuring position for the currents and voltages was in the substation at branch 1. The aim of the neural network remained the same: to locate the earth faults between the nine stations. In this case, it was investigated how an earth-fault-locating neural network reacts if it comes to a more complex grid model.

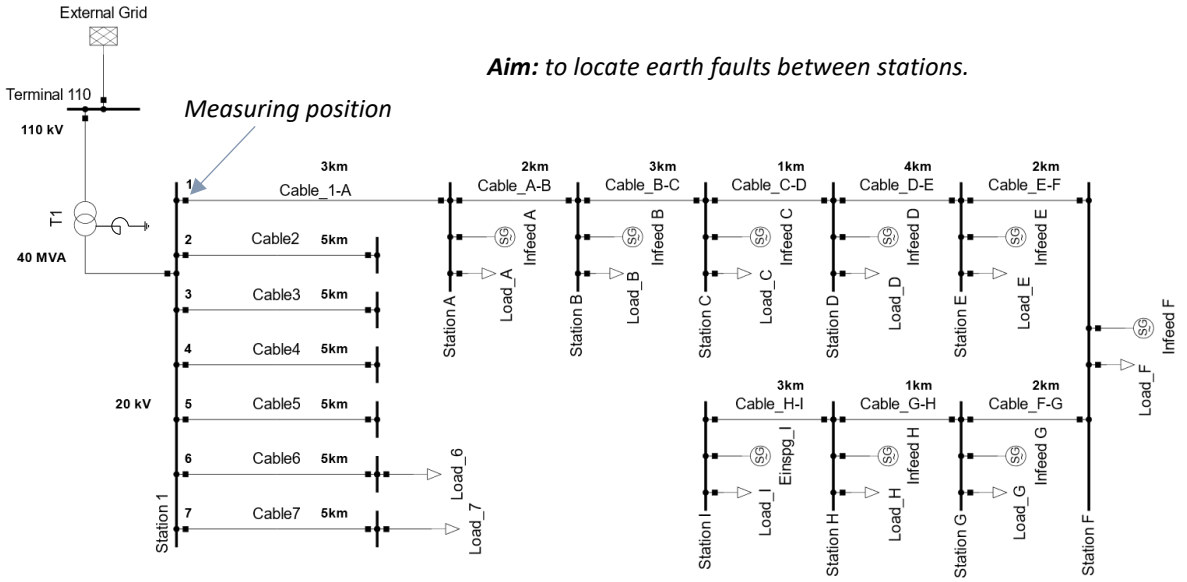


Figure 3-15, earth fault simulation with nine stations on the affected feeder

During the simulation, the following parameters were varied:

	Varied	Steps
Earth fault position	From 10 to 100%	10%
Loads at station A, C, F and I	From 0 to 0.3 MW	0.3 MW
Load at branch 6	From 1 to 5 MW	4 MW

Table 3-10, varied parameter during simulation

The earth fault position relative to the cable's length ranged from 10 to 100% in 10% steps for each cable. The power of the four loads in the local grid stations A, C, F, and I of the earth-faulted branch 1 were varied between 0 to 0.3 MW in 0.3 MW steps. The power of the load of the residual healthy branch 6 ranged between 1 and 5 MW in 4 MW steps.

	Constant
Power infeeds at station A to I	0.15 MW
Loads at station B, D, E, G, H	0.4 MW
Load at feeder 7	5 MW

Table 3-11, constant parameter

The power infeeds at the nine stations were constant at 0.15 MW. Thus, during simulation, if the power of a load in one station is 0 MW, the station fed in 0.15 MW. Otherwise, the station took power from the branch. The loads that were not varied were constant at 0.4 MW. The power of the load at branch 7 was continuously 5 MW.

As before, the generated data samples, including the type of fault, the failure location, and all the parameters, which have changed during the simulation, were stored as CSV files. A total of 2,880 data samples were generated.

3.3.4.3 Model for Earth Fault Localization, 8 Stations, One long Fanout

The third exemplary model consists of eight stations, with the same structure as in the previous grids but including one long fanout of 10 km at station C. The cable lengths at the faulty branch ranged from 1 to 3 km. In this case, the subject of the investigation is how a neural network performs when it comes to parallel cables. It was examined how a long fanout of 10km influences the results of an earth-fault-locating neural network. The earth-fault-locating neural network's aim remains the same to locate the earth faults between the local grid stations at branch 1.

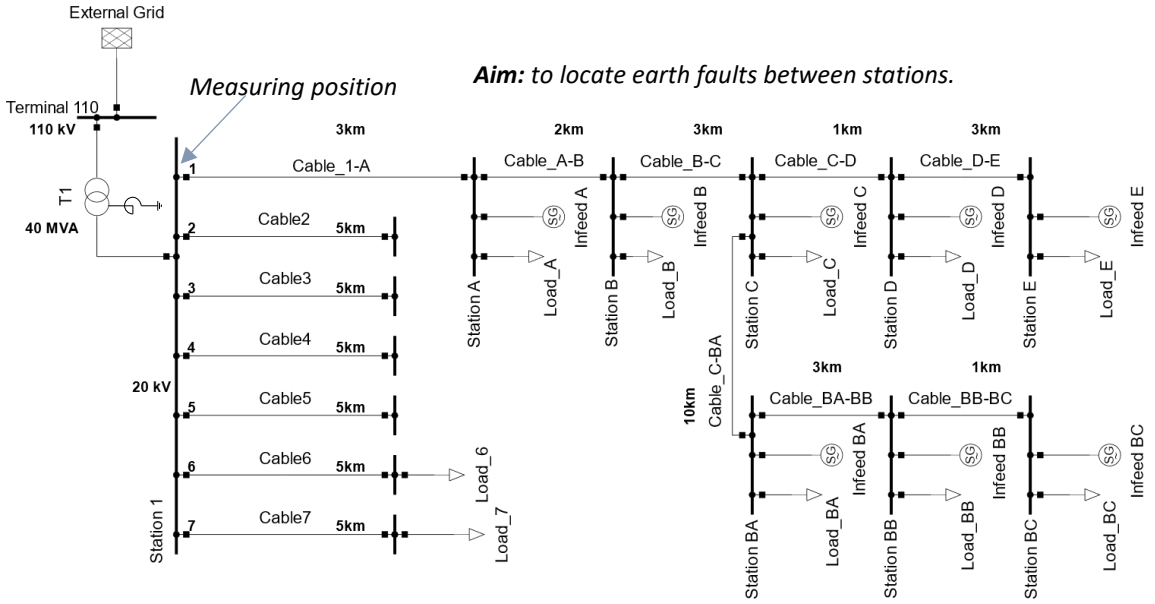


Figure 3-16, earth fault simulation with eight stations on the affected feeder, including one long fanout

During the simulation, the following parameters were varied:

	Varied	Steps
Earth fault position	From 10 to 100%	10%
Loads at station A, C, D and BA	From 0 to 0.3 MW	0.3 MW
Load at branch 6	From 1 to 5 MW	4 MW

Table 3-12, varied parameter during simulation

The earth fault position relative to the cable's length ranged from 10 to 100% in 10% steps for each cable at branch 1. The power of four loads in the local grid stations A, C, D, and BA of the earth fault branch 1 were varied between 0 and 0.3 MW in 0.3 MW steps. The power of the load of the residual healthy branch 6 was varied between 1 and 5 MW in 4 MW steps.

	Constant
Power infeeds at station A to E and BA to BC	0.15 MW
Loads at station B, E, BB, and BC	0.4 MW
Load at branch 7	5 MW

Table 3-13, constant parameter

The generated data samples were stored as CSV files. A total of 2560 data samples were generated.

3.3.4.4 Model for Earth Fault Localization, 8 Stations, One Fanout

The fourth exemplary grid model has the same structure as the model in the previous section, but just with a shorter fanout of 2 km. It was investigated how a neural network performs when the parallel sections are more related to each other, through a shorter fanout of 2 km. Additionally, to see what influence that the number of data samples has on the training results of an earth-fault-locating neural network, the grid model in Figure 3-17 was simulated twice:

- once with earth faults relative to the cable length from 10 to 100% in **10% steps** for each cable at branch 1,
- and once from 10 to 100% of the cable length in **2.5% steps**.

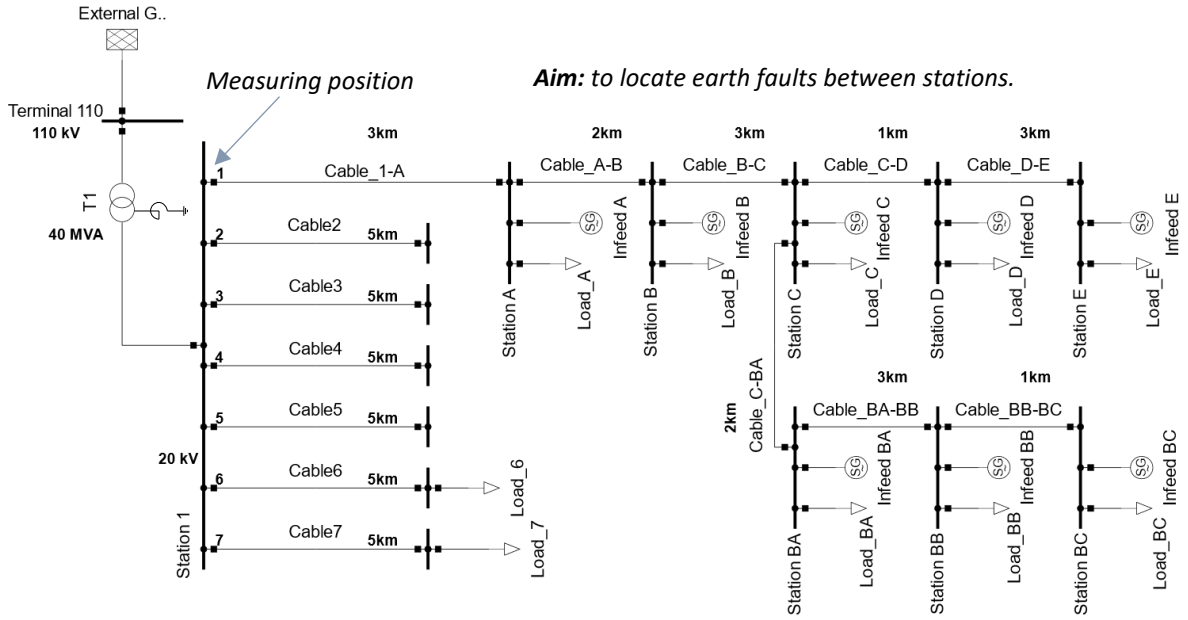


Figure 3-17, earth fault simulation with eight stations on the affected feeder, including a fanout.

Simulations and variations of the grid parameters also took place in the same way as in the previous model. The only difference is the fanout cable of 2 km instead of 10 km.

The varied and constant parameters of the grid remain the same with:

	Varied	Steps
Earth fault position	From 10 to 100%	10%
Loads at station A, C, D and BA	From 0 to 0.3 MW	0.3 MW
Load at feeder 6	From 1 to 5 MW	4 MW

Table 3-14, varied parameter during simulation

	Constant
Power infeeds at stations A to E and BA to BC	0.15 MW
Loads at stations B, E, BB, and BC	0.4 MW
Load at feeder 7	5 MW

Table 3-15, constant parameter

With 10% steps, 2,560 data samples were generated, and with 2.5% steps, 10,240 data samples were generated. The resulting amount of data samples of the second simulation is four times greater than of the first one, with 10% steps. In chapter 0, the different training performances and test results are shown.

Further, this grid model was applied to investigate the limits of the earth-fault-locating neural network in different scenarios. The simulations took place in the same way as before with executed earth faults in 10% steps along the cables. In all three scenarios, the same amount of 2,560 data samples were generated.

In the first scenario, the power infeed at station C was removed to investigate what influence a single power infeed on the performance of a neural network has.

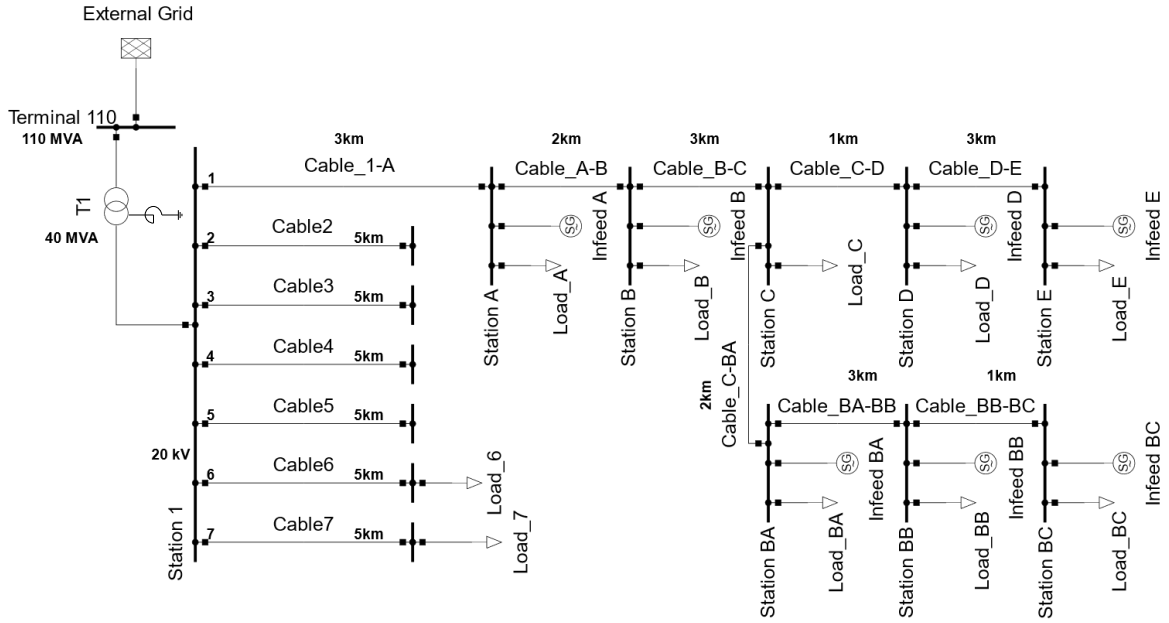


Figure 3-18, earth fault simulation with eight stations on the affected feeder, including a fanout, and without a power infeed in station C

In the second scenario, it was examined how the neural network's performance changes if no power infeeds along the affected branch are included.

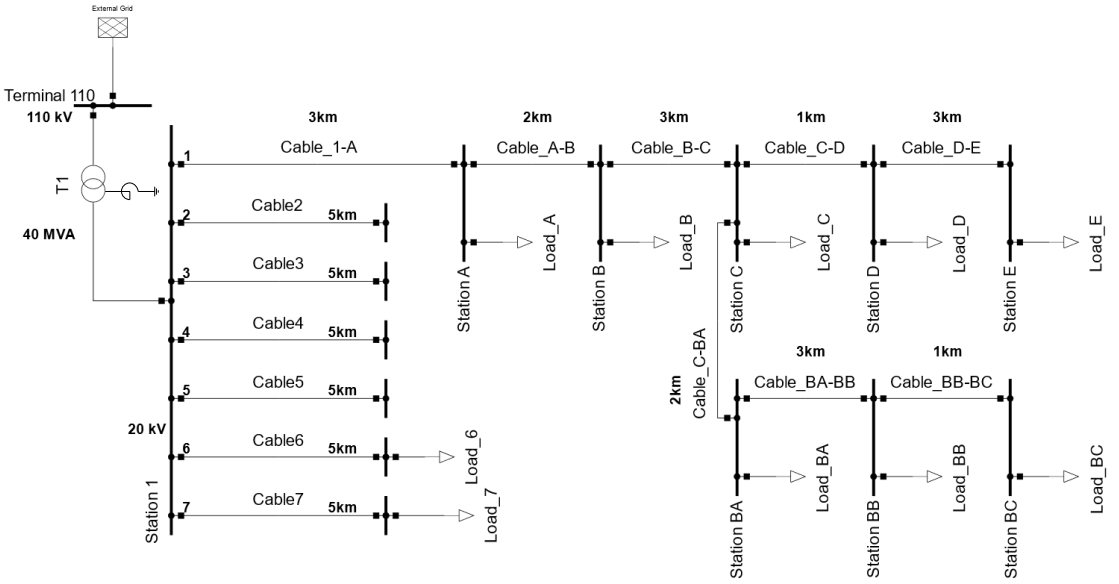


Figure 3-19, earth fault simulation with eight stations on the affected feeder, including a fanout, and no power infeeds

In the third scenario, to find out what influences the loads and power infeeds have, the same grid model was simulated again in open circuit operation. Without any loads and power infeeds along the affected feeder.

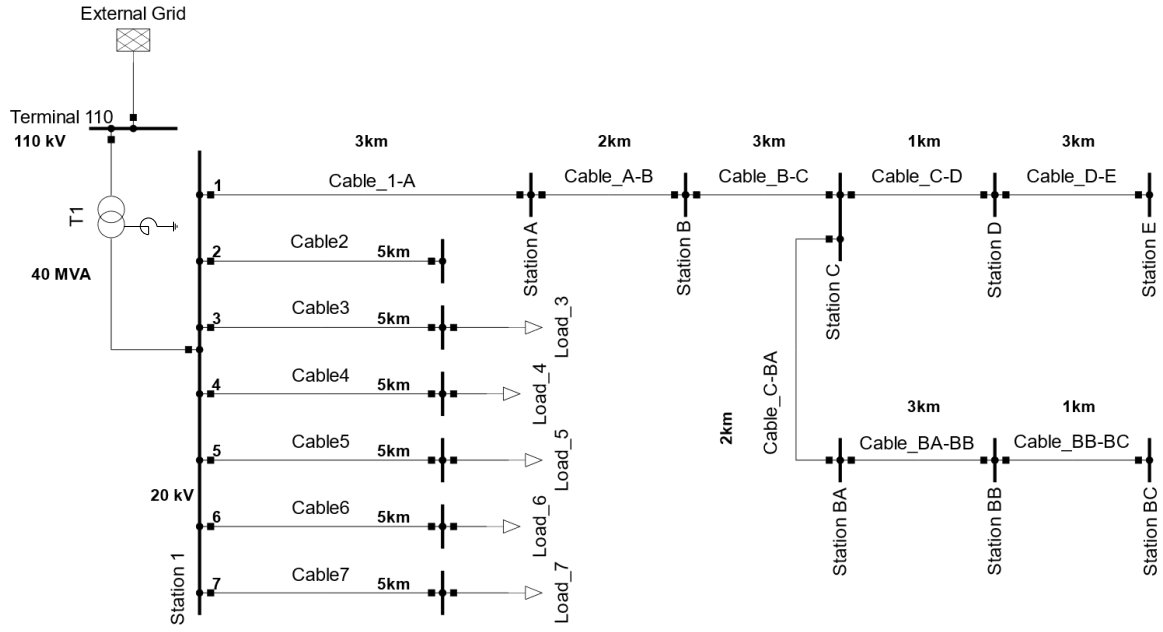


Figure 3-20, earth fault simulation with eight stations on the affected branch, including a fanout and in open circuit operation

3.3.4.5 Model for Earth Fault Localization, 7 Stations with an identically parallel section

The last exemplary model consisted of seven stations at the faulty branch. Hereby, the stations of the parallel section were totally identical, including the same lengths of the cable connections. Also, the variations during the simulation were identical in the parallel section. The emphasis lies in this grid model on the parallel section. In this case, the question was, is it possible to localize earth-faulted sections in a theoretical total identical parallel section?

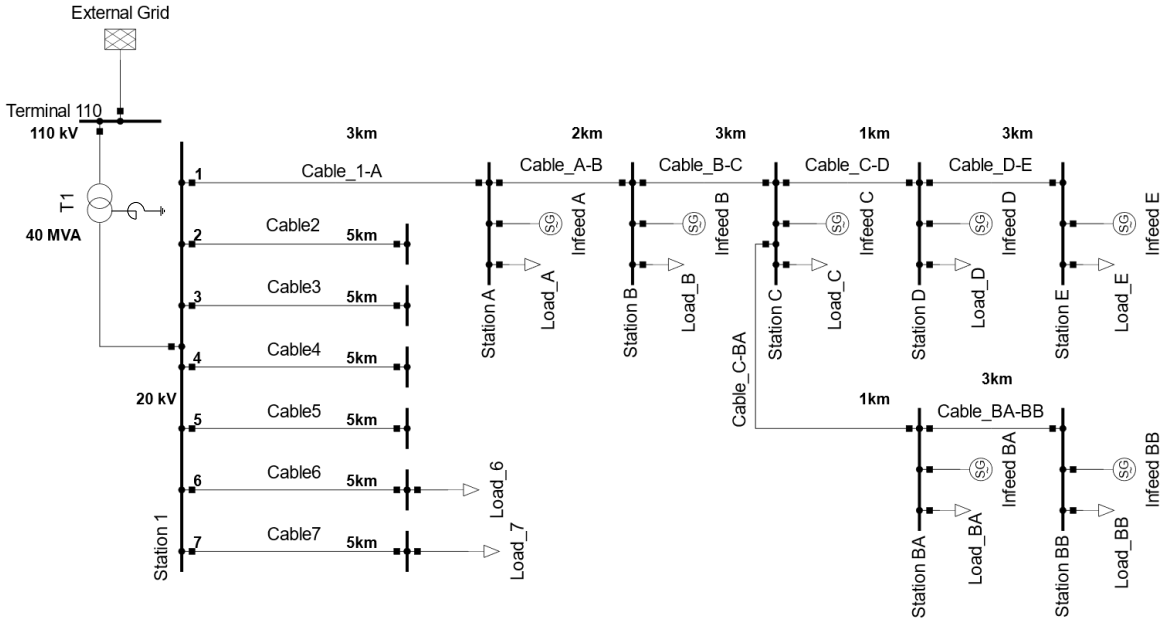


Figure 3-21, earth fault simulation with seven stations on the affected feeder, with a total identically parallel section

During the simulation, the following parameters were varied:

	Variated	Steps
Earth fault position	From 10 to 100%	10%
Loads at station A, C, D and BA	From 0 to 0.3 MW	0.3 MW
Load at branch 6	From 1 to 5 MW	4 MW

Table 3-16, varied parameter during simulation

The earth fault position was relative to the cable's length, from 10% to 100% in 10% steps for each cable. The power of the four loads in stations A, C, D, and BA of the earth-faulted branch 1 were varied between 0 and 0.3 MW in 0.3-MW steps. The power of the loads of the residual healthy branch 6 varied between 1 and 5 MW in 4-MW steps.

	Constant
Power infeeds at station A to E and BA to BB	0.15 MW
Loads at station B, E, and BB	0.4 MW
Load at branch 7	5 MW

Table 3-17, constant parameter

The power infeeds at the seven stations were constant 0.15 MW. The not variated loads were constant 0.4 MW. Thus, during simulation, if the power of a load in one station was 0 MW, the stations fed in 0.15 MW. Otherwise, the station took power from the feeder.

With this grid configuration, 2240 data samples were generated.

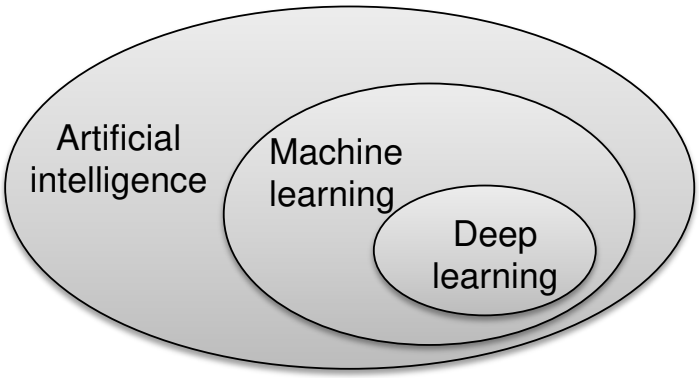
4 Artificial Intelligence, Machine Learning, and Deep Learning

"We call ourselves Homo sapiens—man the wise—because our intelligence is so important to us." [17]

As humans, we used to explore ways of thinking, the process of understanding, and the manipulation of things. With artificial intelligence, we go one step further and try not only to understand how it works but also to create intelligence technical systems. Today, artificial intelligence is comprised of a vast number of applications, from an application for playing chess to image-classification and diagnostic tools. Artificial intelligence has already become a universal field. [17]

4.1 Artificial Intelligence

AI, in general, is a vast field that encompasses machine learning as well as deep learning, as shown in figure 4-1.



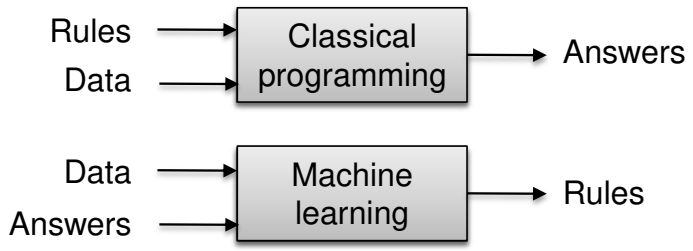
4-1 Artificial intelligence, machine learning, and deep learning, [18]

It includes approaches and applications that do not require any learning and that just seem to be intelligent. Examples of this include early chess programs, which were created by programmers. From the 1950s, when the first AI concepts were published, until the late 1980s, many experts believed human-level AI could be reached with just a large number of explicit rules, handcrafted by programmers. Such approaches are called *symbolic AI*. [18]

Nevertheless, it did not work out for more complex, fuzzy problems, such as speech recognition, language translations, and the classification of images. Machine learning took symbolic AI's place in order to solve these kinds of problems. [18]

4.2 Machine Learning

In classical programming, the rules created and coded by humans are processed with data to get an output. In other words, data is processed according to handcrafted rules, and that process has answers as output (see figure 4-2). A machine learning algorithm outputs rules which are trained on input data as well as the answers expected from the input data. These rules can then be applied to new data. [18]



4-2, classical programming and machine learning, [18]

A machine-learning system can be trained by many examples relevant to a task. During the training process, the ML algorithm maps the input data to their expected and known answers, also called targets, that, in the end, eventually allows the ML system to come up with rules [18]. For example, to classify the type of a short circuit with an ML system, many examples of short circuits already labeled (classified) by humans are needed. The ML system would learn statistical rules for classifying the type of a short circuit for never-seen-before data.

In short, machine learning learns rules from a collection of input-output pairs to predict outputs for new inputs. To be able to do that, three things for machine learning are needed, as described in [18]:

- **Input data:** If the task is short circuit classification, the data points could be measurement data of different types of short circuits.
- **Examples of the expected output:** The measurement data could be labeled as the type of short circuit, for instance, as a 3-phase short circuit.
- **Verification of the algorithm:** It is essential to know the difference between the output of the ML system and its expected output. The measured difference is used as a feedback signal to adjust the ML algorithm. This adjustment is called learning and is executed by the backpropagation algorithm. [18] How the backpropagation algorithm works will be explained in section 0.

Learning in this context means finding appropriate representations for the input data to get the system's output closer to the expected output. Representations are a different way to represent or encode data. Such transformations like linear projections, nonlinear operations, coordinate changes, or decision trees to automatically find more useful representations are contained in all machine-learning algorithms. The algorithms are searching through a predefined set of operations, which is called the hypotheses space. To sum up, machine learning algorithms search for good representations in the input data, within a predefined space of possibilities, the hypotheses space, while using a feedback signal as guidance. [18]

4.3 Deep Learning

This master's thesis will rely on neural networks to classify faults and locate earth-faulted sections.

As illustrated above, deep learning is a subfield of machine learning. It is a particular way to learn representation from input data while using successive layers. The models in deep learning are called neural networks. The number of successive layers in a neural network is referred to as the depth of the neural network. A neural network can be tens to hundreds of successive layers deep. Although the term "neural network" refers to neurobiology, there is no evidence that the human brain implements any of the learning mechanisms used in modern deep-learning neural networks. Instead, the term takes inspiration from our understanding of how the human brain might learn. [18]

The neural network–related core ideas on which deep learning is based originated in the 1950s, but the first breakthrough took decades. The main reason for this delay was the lack of an efficient way to train such large networks. This changed in the mid-1980s, when the backpropagation algorithm was rediscovered by multiple scientists. The backpropagation algorithm uses gradient-descent optimization, which allows the training of chains of parametric operations. In the 1990s, the first practical application of neural networks was used by the United States Postal Service for reading handwritten ZIP codes on mail envelopes. This application came from Bell Labs, especially from Yann LeCun, who combined neural networks and backpropagations to classify handwritten digits. [18]

At the beginning of the 2010s, scientists from the University of Toronto, Montreal, New York City, and the AI Lab ISDIA in Switzerland made important breakthroughs in the field of neural networks. In 2011, IDSIA began to win academic image-classification competitions with trained deep neural networks, which was at the same time the first practical success of modern deep learning. In 2012, a team of scientists from the University of Toronto participated in the large-scale image classification challenge named ImageNet. This challenge consisted of classifying high-resolution color images into 1000 different categories after training on 1.4 million images. The team achieved a top-five accuracy, which was a significant breakthrough at that time. Since then, the ImageNet challenge has been dominated by deep neural networks and was considered to be completed after a participant reached, with a neural network, an accuracy of 96.4% in 2015. After such achievements, deep learning has found applications in many perceptual tasks such as computer vision or natural language processing. [18]

4.3.1 Different Kinds of Deep Learning

Neural networks can be trained in different ways for different applications. Four main methods are presented below.

4.3.1.1 Supervised Learning

In supervised learning, the neural network is trained on input/ output pairs. The neural network is trained on data where the targets for the input data are already known. If the trained neural network is applied to new data, it predicts an output based on the data it was trained on. This method is called supervised

learning and is the most common method in deep learning. [17] [18] In this master's thesis, the focus is on supervised learning.

4.3.1.2 Unsupervised Learning

In unsupervised learning, the neural network learns similarities in the input data without an explicit feedback signal. The most common task in unsupervised learning is clustering—the distinguishing of a possible useful cluster from input samples without any targets. It is often used in data analytics to get a better understanding of a dataset before using supervised learning. [8] [9]

4.3.1.3 Self-Supervised Learning

Self-supervised learning is in some way like supervised learning, but without the part of human labeling. The labels are generated from the input data using heuristic algorithms. An example of self-supervised learning is an autoencoder, the goal of which is to reconstruct the input data and to learn how it is represented in the data. [18]

4.3.1.4 Reinforcement Learning

Reinforcement learning is a method in which the neural network learns on reinforcements of rewards and punishments. The neural network receives information about the environment, and it tends to set actions to increase and maximize the rewards. [18] According to [18], reinforcement learning will soon be used in applications such as robotics and self-driving cars.

4.3.2 How Deep Learning Works

As a subfield of machine learning, deep learning is a mathematical framework for learning representations from input data. Figure 4-3 shows a neural network with four layers for fault classification. Each layer turns its input data into representations, which are increasingly more informative about the final result. In such a classifications task, the output of a neural network is the class with the highest resulting probability in the final layer. With this, the successive layers can be imagined as a multistage information distillation. The data, in an already trained neural network, goes through the layers and comes out in one pot of the different categories. In Figure 4-3, the original input displays three voltages during a 3-phase short circuit. The input goes through successive layers and is classified in the final layer as a 3-phase short circuit.

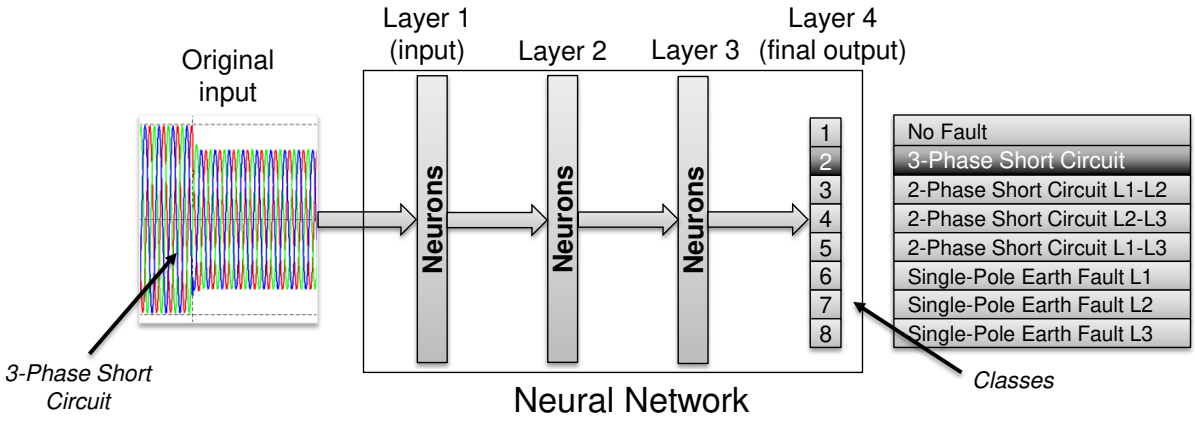


Figure 4-3, Deep neural network for short circuit classification, [18]

Layers in a neural network consist of nodes, learnable parameters, or so-called neurons [19]. The neurons in one layer are connected to the neurons in the previous and following layer. The connections are parameterized by their weights. What layers do to its input data is stored in the layers' weights. These weights are parameterized and adjusted during the training process. [18]

In a neural network, each value of the input data is allocated to one neuron in the input layer. The input layer has no parameter to learn, and therefore, those connections are not weighed. [19] In the case of a 3-phase short circuit, as shown in Figure 4-4, each value of the three measured voltages and currents is connected to one neuron of the input layer. A time series of 4002 steps and six measured quantities results in 24,012 neurons for the input layer. The input layer is followed by two successive, fully connected (dense), hidden layers consisting of four neurons. It is followed by the final layer as the output layer, which contains nine neurons as the number of short-circuit classes is. The output layer, finally, shows the probability distributions for all nine classes. For the 3-phase short circuit example, each type of fault receives a certain percentage of the probability. The highest prediction is the choice of the neural network.

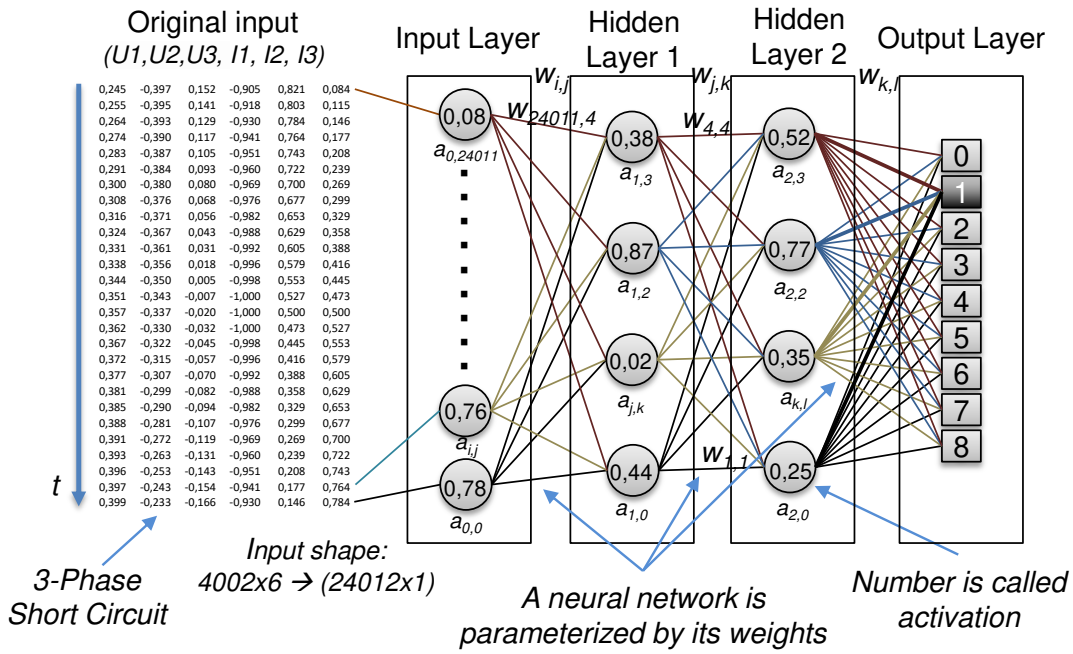


Figure 4-4, the architecture of a neural network

Each neuron in a neural network contains a score resulting from the activation function. The activation function transforms the weighted input connections into the activation, or score, of the neuron. The activation of a neuron can be seen as output.

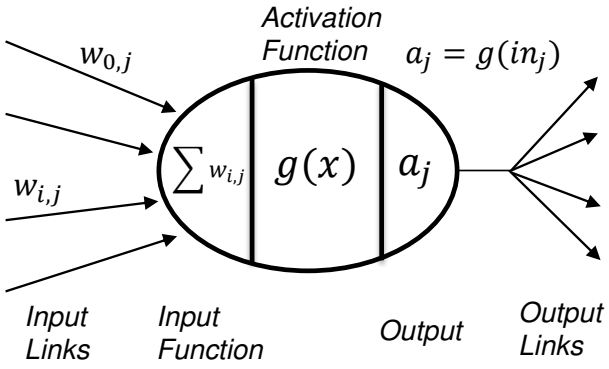


Figure 4-5, a simple mathematical model of a neuron [17]

As shown in Figure 4-5, such a neuron consists basically of

- input links, which result in the sum of all weighted connections,
- an activation function g, like $relu$ or $sigmoid$, and
- an output a of the activation function, called the activation of a neuron.

For a neuron j, k, the output is as follows. [17]

$$a_j = g\left(\sum_{i=0}^{n} w_{i,j} a_i\right) = g(in_j) \qquad 4\text{-}1$$

In other words, the activation of each neuron in a layer is based on the weighted sum of all the activations in the previous layer plus a specific bias. For example, an activation $a_{1,0}$ in the first hidden layer is determined through the following function:

$$a_{1,0} = relu(w_{0,0}a_{0,0} + w_{0,1}a_{0,1} + \cdots + w_{0,24011}a_{0,24011} + b)$$

where $w_{o,n}$ and $a_{o,n}$ are the weighted links and activations from the input layer, and b a bias attribute is. The operation $relu$ (rectified linear unit) is a ramped function that zeros out all negative values. $relu$ is $\max(x, 0)$. With a bias b of $b = -10$, for example, the activation will only be activated when the weighted sum is greater than ten. The bias contributes that only meaningful activations are active. Next to the $relu$ function, many other activation functions exist. One of these is the $sigmoid$ function $\frac{1}{1+e^{-x}}$, which is a scaled and shifted hyperbolic function. However, $relu$ is the most common and popular function in deep learning. Embedded in a matrix and vector form for all activation in a layer: [18]

$$activations\ in\ a\ layer = relu(dot(W, input) + b) \qquad 4\text{-}2$$

The weight matrix W will have the shape (number of neurons of the current layer, number of neurons of the previous layer). The $input$ vector consists of all activations from the last layer. Activations and weights determine activations in the next layer. All activations used in a neural network are differentiable. This advantage will be used to update the weights and biases during the training process. [18]

The single values in the 3-phase short circuit example represent the activations in the input layer. The connections or links from the input layer to the first hidden layer are weighted through the training process and lead to activations of the first hidden layer. The first hidden layer is then connected to the second hidden layer in the same way, and so on. These kinds of neural networks are called feed-forward networks.

4.3.3 Training Process

All the weights in a neural network are adjusted during the training process. These adjustments are the main challenge in deep learning because what a neural network does is stored in its weights. [18] For a fault classifying neural network, a high amount of already labeled training data is needed to start the training.

The training loop is illustrated in Figure 4-6. The original input, for example, a 3-phase short circuit, is fed into the successive layers of the neural network. At the beginning of each training, all weights are initialized randomly. Thus, the neural network outputs some random predictions right from the beginning. The loss function compares these predictions with the correct targets (the desired predictions). The loss score indicates how well the neural network has done on a specific sample. This loss is used as a feedback signal for the optimizer. While using gradient descent, the optimizer adjusts all weights in the correct direction that will lower the loss score for the current example. This training loop is performed to every sample or to every batch of samples in the training data set. Usually, the training process needs to be repeated several times with a whole set of training data to reach a minimum in the loss. Such repetitions are called epochs. [18]

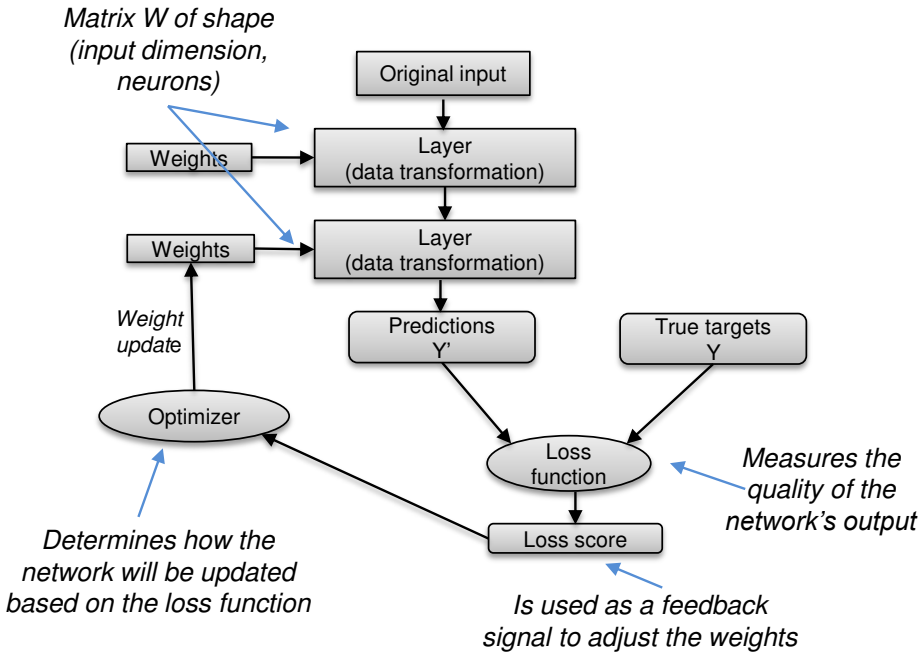


Figure 4-6, training process [18]

Once a network reaches a minimum in loss, and outputs are as close to the correct targets as possible, the network is said to be a trained neural network. The backpropagation algorithm is the central algorithm in the training process and is located in the optimizer and uses gradient descent to update the weights. [18]

4.3.4 Deep Learning with Gradient Descent

The trainable parameters of a neural network are the weights and biases. For a starting point in the training process, they are initialized randomly. While training, these parameters are gradually adjusted, based on the feedback signal.

A batch of samples with their corresponding targets are fed into the network to obtain predictions. The loss function then computes a loss vector about the targets of the network on that batch. The optimizer will then, based on the mismatch of the neural network's predictions and targets, update all weights in it in a way that will lower the loss for the current batch. The gradient descent algorithm is implemented in the optimizer. The negative gradient shows the direction of the steepest descent in a local point of a multidimensional function. For minimizing the loss, the optimizer moves stepwise in the direction of the negative gradient. This step is called the learning rate of the training process. The following example shows how the gradient descent and the backpropagation algorithm are applied in a training loop. [18]

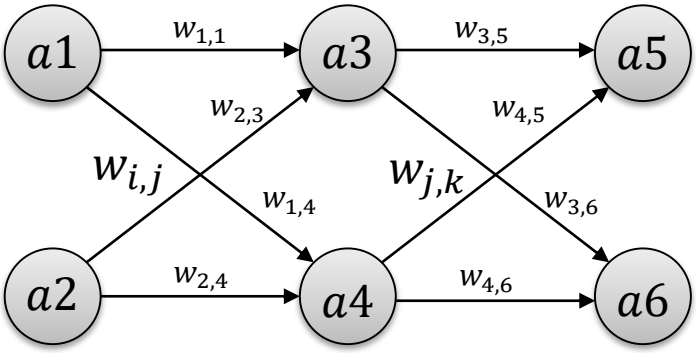


Figure 4-7, A neural network with two inputs, one hidden layer consisting of two neurons, and an output layer.

The neural network shown in Figure 4-7 has two inputs, one hidden layer consisting of two neurons, and two neurons in the output layer. This example does not represent a real-world neural network, but through this, the training process can be explained in detail. The output function y' is parameterized by the weights W. Assuming x_1 and x_2 are the two inputs, the activations in the input layer are set to $(a_1, a_2) = (x_1, x_2)$. The output at neuron a_5 is determined by the following function, where g is the activation function:

$$a_5 = g(w_{0,5} + w_{3,5}a_3 + w_{4,5}a_4)$$

The activations a_3 and a_4 are in the hidden layer, which is connected to the input layer.

$$a_5 = g(w_{0,5} + w_{3,5}\, g(w_{0,3} + w_{1,3}a_1 + w_{2,3}a_2) + w_{4,5}\, g\big(w_{0,4} + w_{1,4}a_1 + w_{2,4}a_2\big))$$

The input neurons are set to x_1, x_2 and lead to the following:

$$a_5 = g(w_{0,5} + w_{3,5}\, g(w_{0,3} + w_{1,3}x_1 + w_{2,3}x_2) + w_{4,5}\, g\big(w_{0,4} + w_{1,4}x_1 + w_{2,4}x_2\big))$$

Now, the output neuron is expressed as a function of inputs and weights. Such functions are differentiable; accordingly, it is possible to use the gradient descent loss-minimization method to update the weights. The neural network above has multiple outputs (two different categories/classes), which return as output a vector function $Y'(W)$, precisely the vector $[a_5, a_6]$, with the same dimension as the expected output (target) $Y = [y_1, y_2]$. The loss is the distance between the predicted output and the expected output of the neural network. [17] The gradient of the loss function for the kth neuron in the output layer with respect to any weight w, as it is described in [17].

$$\frac{\partial}{\partial w} Loss(W) = \frac{\partial}{\partial w}|Y - Y'|^2 = \frac{\partial}{\partial w}\sum_k (y_k - a_k)^2 = \sum_k \frac{\partial}{\partial w}(y_k - a_k)^2 \qquad 4\text{-}3$$

With k as an index for the neurons in the output layer. As shown through the final summation, each term represents the gradient for the kth. For the final weight update, each gradient contribution must be added. Thus, the m output learning problem can be broken down into m learning problems. [17]

As mentioned above, the idea is to use the negative gradient descent to go a step α (defined as the learning rate) downhill, to reach the global minimum. The weights linked from the neurons in the last

hidden layer to the neurons in the output layer are updated, while training, as follows, with the indexes i, j, and k for the neurons in the input-, hidden-, and output layer:

$$w_{j,k}^{new} = w_{j,k}^{old} - \alpha \frac{\partial}{\partial w_{j,k}} Loss_k \qquad 4\text{-}4$$

For the connections between the input neurons and the neurons in the hidden layer, the error needs to be backpropagated. Each hidden neuron j is responsible for some fraction of the error in each of the output neurons. The error values are propagated back to provide error values for the hidden layers.

$$w_{i,j}^{new} = w_{i,j}^{old} - \alpha \frac{\partial}{\partial w_{i,j}} Loss_k \qquad 4\text{-}5$$

As in [17], the application of the backpropagation process can be summarized as follows:

- Calculate the loss gradient for the neurons in the output layer
- Iterate the following steps for each layer, beginning with the output layer to the first hidden layer:
    - Backpropagate the gradient loss to the previous layer
    - Update all weights between those layers

In short, deep learning means minimizing a loss function. The goal is to find the global minimum for the best result by taking small steps downhill. Hereby, the learning rate α needs to be a reasonable value. When the learning rate is too low, it is easy to get stuck in a local minimum. Besides, a lower learning rate requires more computing capacity due to longer convergence time. If the rate is too high, the updates may end up taking completely random locations in the multidimensional function. [18]

4.3.4.1 Backpropagation Example

This section shows how the gradient descent algorithm is applied to a reduced and simplified neural network. The following example is held on [17] and [20]. In order to simplify the calculations, the example is calculated without biases. The neural network will be the same as already introduced in Figure 4-7.

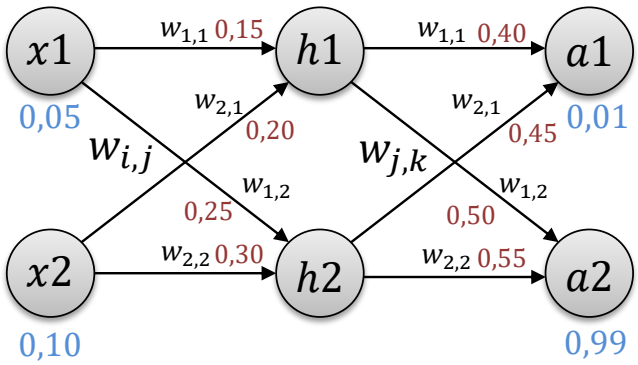


Figure 4-8, backpropagation example

In general, all weights are initially assigned by a random function; here, the weights are chosen by hand. The given input and weight values are shown in Figure 4-8. As mentioned above, the training process aims to minimize the loss function through backpropagation. The wanted output (target) is for $a_1 = 0.01$ and for $a_2 = 0.99$.

As activation function $g(x)$, the sigmoid function is used.

$$g(x) = \frac{1}{1 + e^{-x}} \qquad 4\text{-}6$$

By using the function 4-1, the output of the hidden neurons h_1 and h_2 is calculated as follows to:

$$h_1 = g\left(\sum_{i=0}^{n} w_{i,j} a_i\right) = g(w_{1.1} x_1 + w_{2.1} x_2) = 0.506874567,$$

$$h_2 = 0.510623401$$

for the given input values in Figure 4-8. The same process is applied for the output neurons a_1 and $a_{2,}$ resulting in:

$$a_1 = 0.606477732,$$

$$a_2 = 0.630480835.$$

Next, the loss is determined by using the squared error function 4-3.

$$Loss(W) = |Y - Y'|^2 = \sum_k (y_k - a_k)^2$$

$$Loss_{a1} = (0.01 - 0.606477732)^2 = 0.355785685$$

$$Loss_{a2} = (0.99 - 0.630480835)^2 = 0.12925403$$

To receive the total loss, $Loss_{a1}$ and $Loss_{a2}$ are added up:

$$Loss = Loss_{a1} + Loss_{a2} = 0.485039715$$

Starting with the weights connected with the output layer, the backpropagation algorithm is applied using equation 4-4, with a learning rate of 0.5.

$$w_{j,k}^{new} = w_{j,k}^{old} - \alpha \frac{\partial}{\partial w_{j,k}} Loss =$$

$$w_{1,1}^{new} = w_{1,1}^{old} - \alpha\big(-2(y_k - a_k) a_k (1 - a_k) h_j\big)$$

$$= 0.4 - 0.5 \cdot 0.144314146 = 0.327842927$$

$$w_{1,2}^{new} = 0.45754475$$

$$w_{2,1}^{new} = 0.377309255$$

$$w_{2,2}^{new} = 0.507230752$$

Now, the adjustments for these weights are determined. Next, equation 4-5 is applied to calculate the weights between the input- and hidden layers. The weights are only updated when all weights have been recalculated. Thus, for continuing the backpropagation, the original weights are used.

$$w_{i,j}^{new} = w_{i,j}^{old} - \alpha \frac{\partial}{\partial w_{i,j}} Loss$$

$$w_{1,1}^{new} = w_{1,1}^{old} - \alpha \left[\sum_{k=1}^{2} -2(y_k - a_k)a_k(1 - a_k)\, w_{j,k}\right] \cdot a_j(1 - a_j)a_i$$

$$= 0.15 - 0.5 \cdot 0.000376511 = 0.149811745$$

$$w_{1,2}^{new} = 0.249775187$$

$$w_{2,1}^{new} = 0.199623489$$

$$w_{2,2}^{new} = 0.299550373$$

After all new weights are determined, the new total loss of the neural network can be calculated. In this case, the new total loss is as follows.

$$Loss_{new} = 0.471622467$$

The first total loss was 0.485039715. After the weight adjustment, the total loss decreased to 0.471622467. By this example, it was shown how a weight update in a neural network works. If the algorithm is repeated a sufficient number of times, the total loss would finally reach a minimum. When scaled up in layers and neurons, such a neural network is able to learn patterns in large data sets to classify, for example, short circuits.

4.3.5 Training, Validation, and Test Sets

To be able to evaluate the training process and the neural network itself, before training, the data is split into three parts: training, validation, and test data sets. While training, the neural network is trained on the training data and evaluated by the validation data set. Once it is trained, it is finally evaluated on a test data set that the neural network has never seen before. [18] It is common to split the data, as shown in Figure 4-9, and use 70% for training and 15% for validation and 15% for testing.

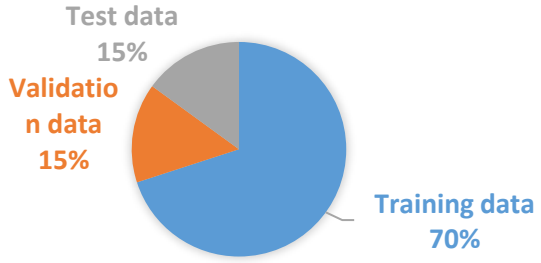


Figure 4-9, training, validation, and test set

4.3.6 Overfitting and Underfitting

The central obstacle in machine learning is overfitting. A fundamental issue is the tension between optimization and generalization. Optimization refers to the process of adjusting a model based on the training data. In contrast, generalization refers to how well the model performs on never-before-seen data. The aim is to reach a useful generalization based on the training data. Overfitting happens when the model is doing well on the training data but worse on the validation data. [18]

Figure 4-10 shows a training example of a neural network. On the left, the loss of the training and the validation data set, while training. On the right, vice versa the loss, the accuracy of the training and validation data. For example, the accuracy (in percent) can be used as a guideline of how many samples were classified correctly. For the whole data, the training loop was repeated 25 times, counted in epochs.

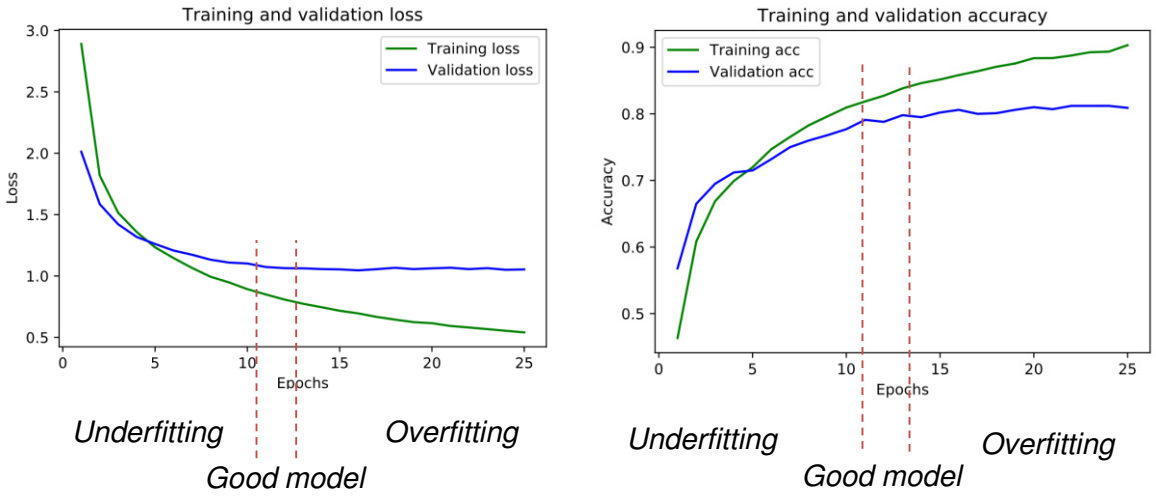


Figure 4-10, overfitting and underfitting

Since all weights are initialized randomly, the loss is very high in the beginning, but rapidly decreases after a few epochs. While the training and validation loss are correlating, the neural network is said to be underfitted. In other words, the neural network has not learned all relevant patterns in the data yet. But at a certain point, the neural network starts to only improve on the training data and do worse or stagnate on the validation data. The neural network starts to overfit. Thus, the neural network learns patterns, which are irrelevant when it comes to new data. [18]

To prevent under- and overfitting, it is important that the training process is stopped at the right time. For example, in Figure 4-10 that would be between epoch eleven and thirteen. Otherwise, the neural network can still make progress or is too adjusted to the training data.

Next to the right stopping of the training, the best solution to prevent overfitting is to use more data. The more training data, the better the neural network will generalize. When no more data is available, several techniques can be applied to prevent overfitting. These techniques are called regularizations. Below, the most common regularization methods are introduced, as in [18].

4.3.6.1 Reduction of the neural network's capacity

The capacity refers to the number of layers and neurons used in a neural network. The more that are used, the higher the capacity. This includes the memorization capacity. Reducing this capacity is one of the simplest ways to prevent overfitting. [18]

On the other hand, if the neural network's capacities are too low, it won't be able to learn correctly and will eventually underfit. Since there is no formula or equation to determine the capacity of a neural network, one should start the training with a small network and increase the number of layers and neurons step-by-step until the results show a proper generalization. [18]

4.3.6.2 Weight regularization

When applied, weight regularizations ensure that weight values are given less entropy, which mitigates overfitting. The weights must take small values, which makes the weight distribution more regular. The loss function adds a specific cost for large weights. As described in [18], there are two common weight-regularization methods.

- L1 regularization. The cost is added proportionally to the absolute value of the weight coefficient. This is known as the L1 norm of the weights.
- L2 regularization. The cost is added proportionally to the square value of the weight coefficients. This is known as the L2 norm of the weights. In the context of neural networks, the L2 regularization is also called weight decay.

4.3.6.3 Dropout

One of the most commonly used and effective techniques to prevent overfitting is the dropout method. The method is applied to one or more layers in the neural network. By using dropout, a predetermined percentage (dropout rate) of neurons is dropped out during training. Commonly, the chosen dropout rate is between 0.2 to 0.5. By a dropout rate of 0.2, 20% of the neurons on the applied layer are zeroed out. Dropout forces the neural network to learn more robust representations and helps to improve generalization. [18]

4.3.7 Keras

Keras is an open-source deep learning library from Google, written in Python. The library was created by François Chollet in 2015 and quickly became the world's most used library for deep learning. [18] It is famous for its user-friendly and modular structure. Keras provides a uniform API (application programming interface) for various backends, like Tensorflow, Microsoft Cognitive Toolkit, and Theano. [18] In this master's thesis, Tensorflow with Keras was used to build neural networks. In the appendix, a code example of a neural network is shown in detail.

4.3.8 Data Preparation

Since all fault samples were stored in CSV files, it was needed to prepare the data samples in a way that the samples can be fed into neural networks. All samples were labeled through their CSV file names. The contained data and their corresponding labels had to be stored in separate arrays so that the samples could be fed into neural networks. The data preparation followed the following procedure.

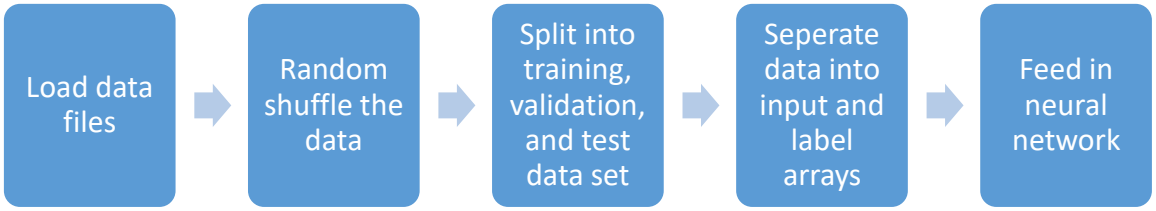


Figure 4-11, data preparation

First, all data files were loaded. Second, the data files needed to be randomly shuffled to avoid any misleading structure in the data, which can lead to wrong representations in the training data. This also makes sure that the training, validation, and test data sets have near the same relative number of different labels in their data sets. After it, the data was split into training 70%, validation 15%, and test 15% data sets.

In the next step, the individual CSV files were read and written into arrays. In the same step, their corresponding labels were written into a separate array. This approach was applied to every single data file. In the end, each sample, in the input array, corresponded to its respective target in the label array. For example, the 10th sample in the input array corresponded to the 10th label in the label array. How this was done exactly can be seen in the appendix. After this procedure, the data was ready to be fed into neural networks.

5 Results

Neural networks were trained to classify faults and to locate earth-faulted sections in compensated medium voltage networks. The simulated and generated data, shown in chapter 3, served as a data basis for the development of neural networks.

5.1 Fault Classification

The first task was to classify different kinds of faults via a neural network. The 20kV compensated medium voltage grid shown in Figure 5-1 was the exemplary grid used for the fault classification task. As introduced in subchapter 3.3.2, the following faults were simulated along cable1 with the measuring position in the substation.

Simulated Faults:

- Earth fault L3
- Earth fault L2
- Earth fault L1
- 2-phase short circuit L1–L3
- 2-phase short circuit L2–L3
- 2-phase short circuit L1–L2
- 3-phase short circuit

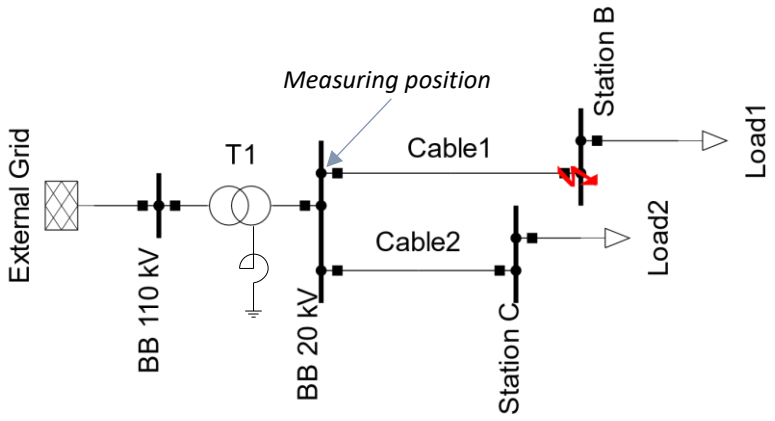


Figure 5-1, a grid for fault simulation

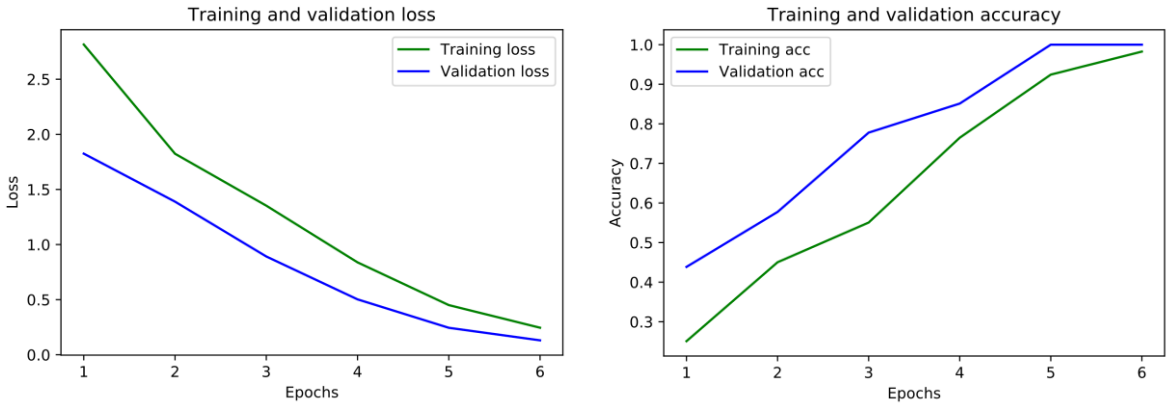
5.1.1 Results

Before the training took place, the 3,360 data samples were split into training data (70%), validation data (15%), and a test data set (15%). As mentioned above, the neural network was trained on the training data and evaluated on the validation data. The test data can show how well a neural network is generalized to data in which it is not trained.

Figure 5-2 shows the training results of the fault-classifying neural network. The left diagram displays the loss of the training and validation data. The curves indicate a good generalization since they decrease almost parallel to a minimum after six epochs.

The right diagram displays the training and validation accuracy. The accuracy of the training and validation data shows to what extent the neural network made correct predictions during the training. After six epochs, the neural network could predict with 100% accuracy regarding the validation data set. The lines are increasing in the same way the loss decreases, indicating a proper generalization. For the remaining never-before-seen 15% test data set (504 data samples), the neural reached 100% accuracy. It is another indicator that the neural network was well-trained. These results show that a neural network is capable of classifying faults.

The difference between the training and validation loss is that the training loss is the average loss of all training samples running through the training loop during one epoch. The validation loss is the average loss of all validation samples measured at the end of one epoch. In the beginning of an epoch, the neural network is less trained than in the end of an epoch. That is why the validation loss is lower than the training loss. The same goes, just vice versa, for the training and validation accuracy.



Accuracy of 100%, on the never-before-seen test data set.

Figure 5-2, training and validation loss (left), training and validation accuracy (right)

A neural network for classification puts out probabilities for each class. The class with the highest probability is the neural network's choice or prediction for an input. Figure 5-3 shows the probability distribution of the neural network over the different classes of 100 randomly selected test samples. The entire test data set contains 504 samples but would not fit on the paper. Therefore 100 samples were chosen at random. The 100 randomly selected test samples were compared to the whole test samples to avoid any wrong conclusions out of the 100 test samples.

In Figure 5-3, the red x in the diagram show which type of fault received the highest probability from the neural network. The white dots show the actual type of the fault. In this case, all faults are predicted correctly. On the y-axis, the different faults are listed. The x-axis shows the test samples, which are independent of each other. The test samples are sorted according to the type of fault.

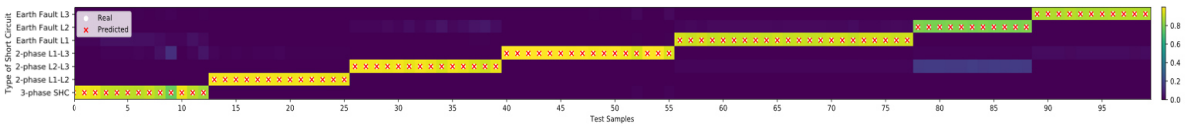


Figure 5-3, diagram for the probability distribution on 100 randomly chosen test samples

This visualization (Figure 5-3) allows to evaluate the neural network's quality. The color bar on the right enables a correlation between the color and probability distribution for each sample. The more yellow, the higher the predicted probability for a certain type of fault. The darker, the lower the probability.

The diagram shows the high quality of the predictions of the neural network due to the correct and highly-rated predictions. All faults were correctly predicted. The single predictions received at least a probability of 75%, which indicates such a high quality of the results. These results lead to the conclusion that neural networks are capable of fault classification.

5.2 Earth Fault Localization

As described above, the goal for the earth-fault-locating neural networks was to locate earth fault positions between local grid stations. Investigated were different grid topologies that differed in complexity, to find out limits of such earth-fault-locating neural networks.

5.2.1 Model 1 - 4 Stations

A neural network was trained with the aim to locate earth-faulted sections along an affected branch. The first grid model for the earth-fault-locating task, as introduced in subchapter 3.3.4.1, consisted of four local grid station connected with cables of the same length.

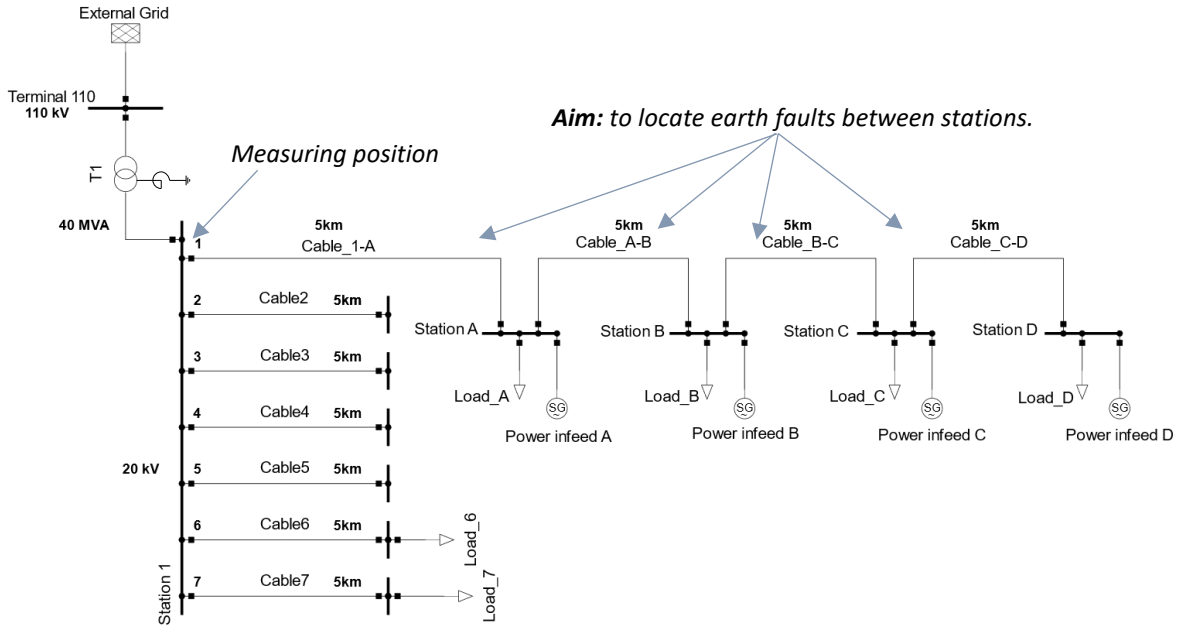
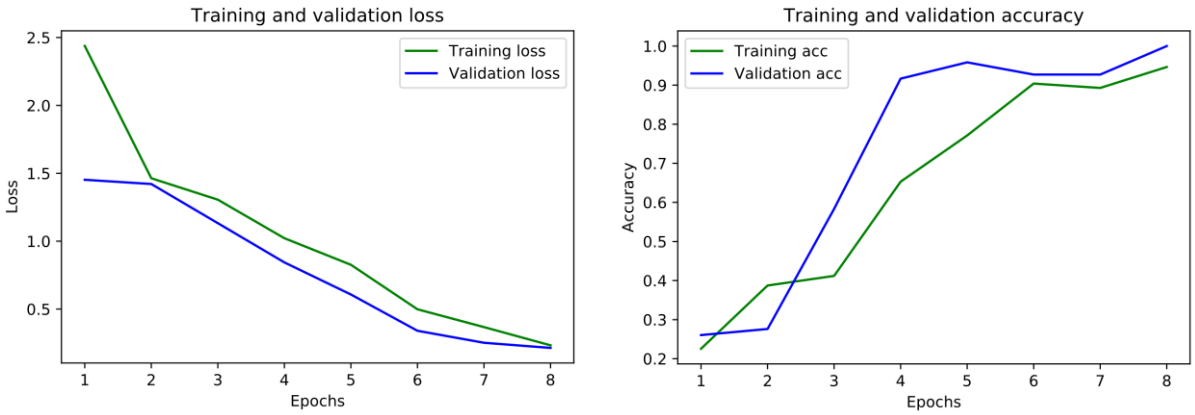


Figure 5-4, earth fault simulation with four stations on the affected feeder.

5.2.1.1 Results

The generated data was split into 70% training data, 15% validation data, and 15% test data. The training performances are shown in Figure 5-5. On the left side, the training and validation loss is shown, and on the right side, the training and validation accuracy. The curves in the loss and in the accuracy diagram are decreasing and increasing almost in the same way, which is a good sign for a proper generalization.



Accuracy of 100%, on the never-before-seen test data set.

Figure 5-5, training and validation loss (left), training and validation accuracy (right)

Through random initialization at the beginning of the training process, the loss is very high, and the accuracy is very low. After eight epochs of training, the process was stopped at an optimum, where the neural network was not underfitting or overfitting. The accuracy of the validation data reached at the end was 100% and indicated such an optimum. With the test data set, an accuracy of 100% could be achieved.

In order to assess the strengths and weaknesses of the earth-fault-locating neural network, the probability distribution of the neural network's outcome was visualized. Figure 5-6 shows 100 test samples. The 100 test samples were selected from the test data set by a randomizing function. The entire test data set would not fit on the paper size. It was ascertained that the random test samples do not change the conclusion of the results. The test samples were sorted by their simulation position on the cables. Thus, it was possible to identify how the neural networks react at transition points.

On the y-axes of the diagram, the single-cable sections are listed. On the bottom, the first cable of the faulty branch, and on the top the last cable. The x-axis shows the test samples, which are independent of each other. The background color indicates the probability distribution for each cable on a test sample. The more yellow it is, the higher the predicted probability of an earth fault along a cable. The darker it is, the lower the predicted probability of an earth fault. The sum of the probability distribution for each test sample is one.

The white dots indicate where the earth faults are actually located, and the red x's indicate where the earth faults were predicted to be with the highest probability by the neural network. In this case, the predicted cable is always the one with the highest probability. In general, this type of diagram gives a good overview of how well a neural network works in a particular grid.

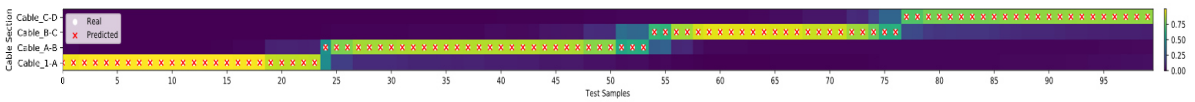


Figure 5-6, diagram for the probability distribution on 100 randomly chosen test samples

Figure 5-6 shows that all earth fault locations were predicted correctly. At the beginning (Cable_1-A), the earth faults were predicted with 100% accuracy. At the transition points from one cable section to another, the probability of the correct faulty cable decreased to around 50 to 70%. In the middle of the cable sections, the probability output of the neural network for the correct prediction increased again. In summary, the diagram shows that neural networks can localize earth faults.

5.2.1.2 Conclusion

The diagram points out to what extent of accuracy the neural network is working for never-before-seen data. The visualization shows that the trained neural network can be a useful tool to locate the area or bound the area of the fault, although there are some uncertainties at the transition points.

5.2.2 Model 2 - 9 Stations

The second grid model for the earth-fault-locating task is shown in Figure 5-7, which consists of nine local grid stations and cable connections of different lengths. The aim of the earth-fault-locating neural networks remained the same, that is, to locate the earth faults between the nine stations. The aim of this grid model was to investigate the limits of the earth-fault-locating neural network by using a more complex grid model.

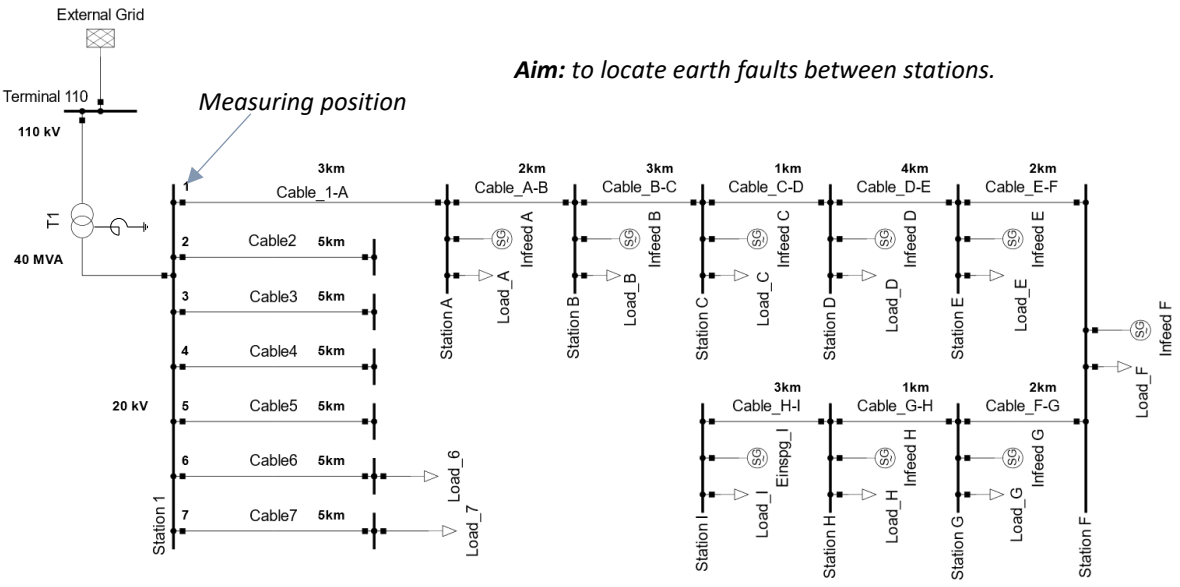
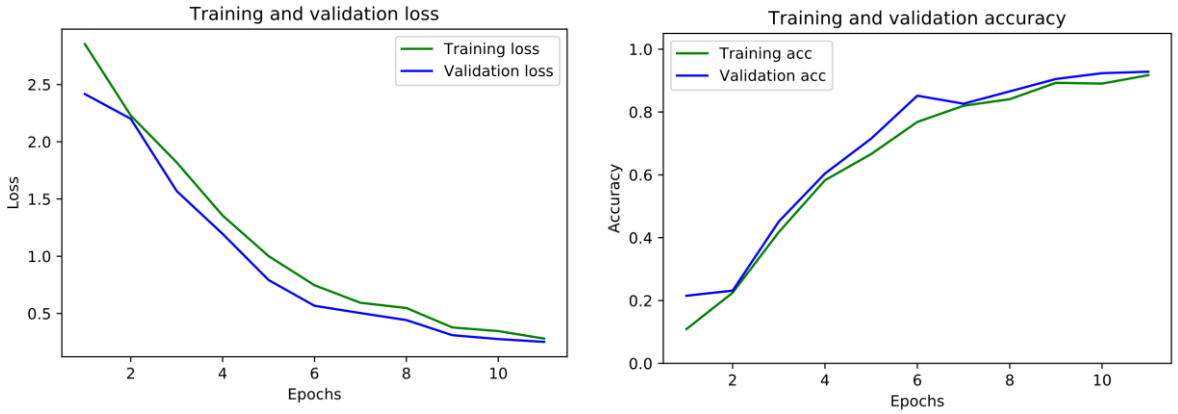


Figure 5-7, earth fault simulation with nine stations on the affected feeder

5.2.2.1 Results

The 2,880 data samples are split into training data (70%), validation data (15%), and test data (15%). In Figure 5-8, the training performances are shown. The training and validation loss and accuracy are correlated to each other, which indicates a good generalization of the neural network.



Accuracy of 93% , on the never-before-seen test data set.

Figure 5-8, training and validation loss (left), training and validation accuracy (right)

The training stopped after eleven epochs. Thus, the neural network was not overfitting. For the never-before-seen test data set, an accuracy of 93% was achieved. The accuracies of the test data set and the validation data set were nearly identical. These values also indicate a proper generalization of the neural network.

Figure 5-9 displays the predicted probability distribution over 100 test samples. The cable sections are on the y-axis, and the test samples are on the x-axis. The red x show the predicted earth fault locations, and the white dots show the actual earth fault locations.

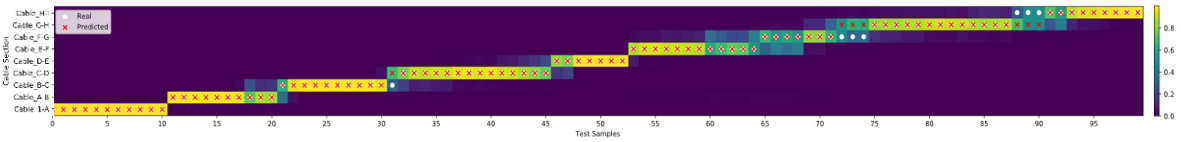


Figure 5-9, diagram for the probability distribution on 100 randomly chosen test samples

The visualization shows a high quality of the predictions at the beginning (first cable). “High quality” in this context means that the predicted earth fault positions are predicted correctly with a high probability, and incorrectly with a lower probability. At the transition points, that quality is lower; it is significantly lower the farther away the earth faults from the substation are. In sum, the figure shows with an accuracy of 93% that the neural network can classify earth-faulted cables. At the transition points, the neural network clearly marks the earth-faulted area.

5.2.2.2 Conclusion

Compared to the previous network model, it has been shown that the neural network can classify earth-faulted cable sections with high accuracy, even if there were more grid stations and different cable lengths.

5.2.3 Model 3 - 8 Stations, One Long Fanout

In this grid model, it was investigated how a neural network reacts when it comes to a parallel section. Therefore, two different configurations for the following grid topology were simulated: one grid model with a longer cable fanout and one with a shorter cable fanout. The following grid model consists of one long fanout of 10 km. The structure of the grid stations is the same as in the previous model. The subject of this investigation is how a neural network performs when it comes to parallel cables. The grid has the same measuring position and the same aim for the neural network to locate earth faults between the local grid stations at branch 1.

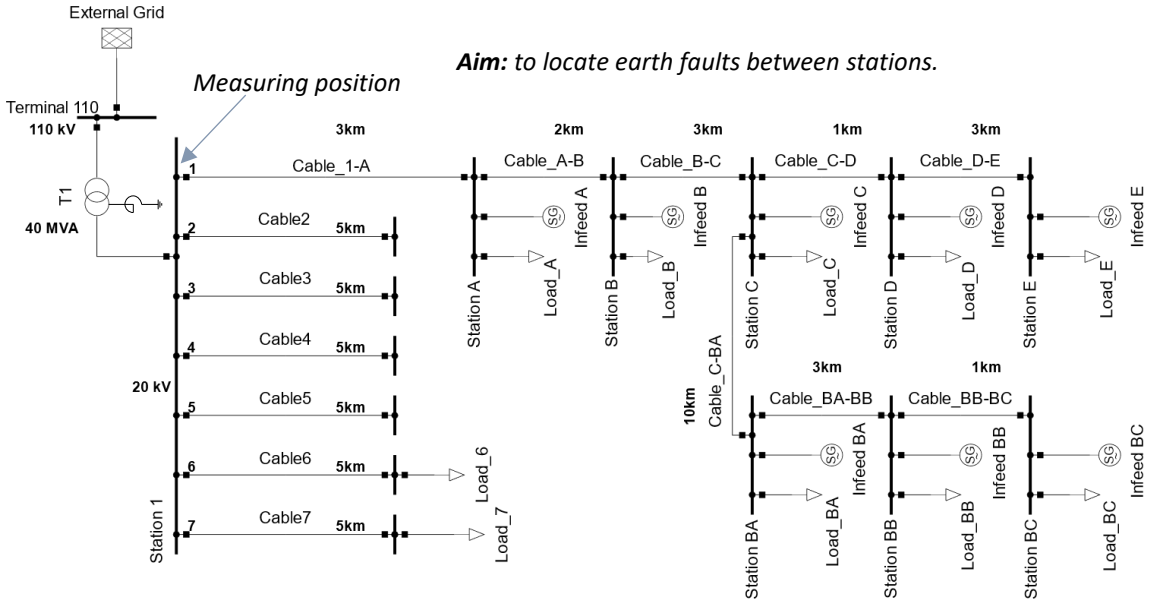
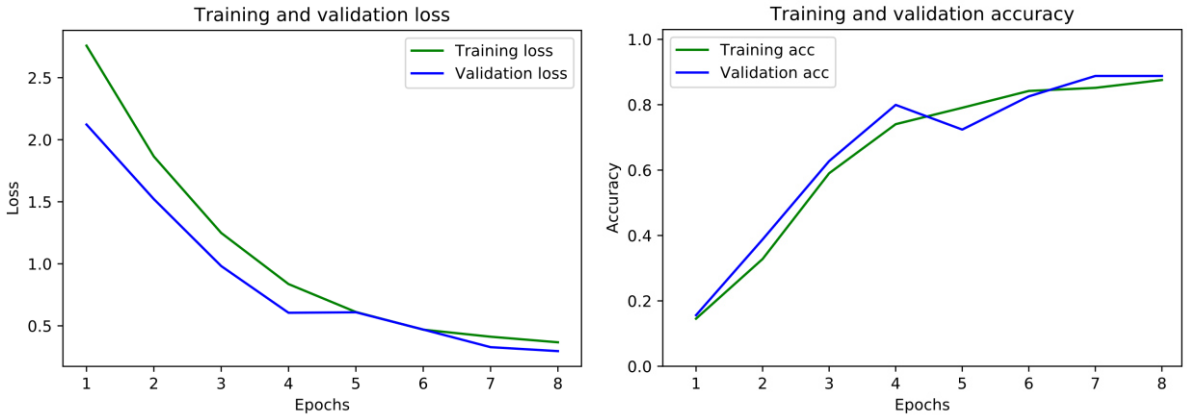


Figure 5-10, earth fault simulation with eight stations on the affected feeder, including one long fanout

5.2.3.1 Results

The 2560 data samples were split into training data (70%), validation data (15%), and test data (15%). The training performances show a satisfying generalization. The training of the neural network stopped after eight epochs.



Accuracy of 88%, on the never-before-seen test data set.

Figure 5-11, training and validation loss (left), training and validation accuracy (right)

An accuracy of 88% was achieved for the test data set. The 88% of the test dataset are very close to the accuracy of the validation data, reached after eight epochs. It is a sign that the training was stopped in the right position.

The visualization of the predicted probability distribution over the 100 test samples is shown in Figure 5-12. At the first three cable sections, the neural network predicted, with high quality, the correct earth-faulted sections.

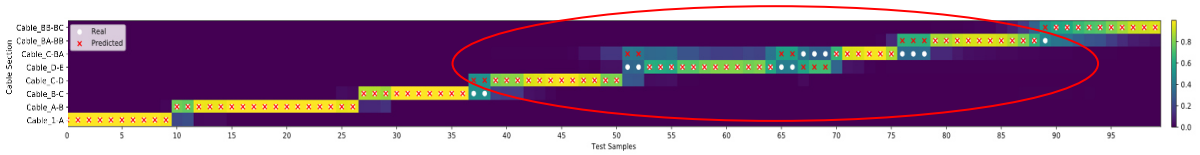


Figure 5-12, diagram for the probability distribution on 100 randomly chosen test samples

When it comes to the parallel section, the neural network predicted most of the earth fault locations correctly but with a lower probability, especially at the transition points in the parallel section. At the end, when one cable is running out, the probability for the right predictions increased again. Even when the earth fault location is predicted wrongly, the neural network is not far away with its prediction from the actual earth fault section.

5.2.3.2 Conclusion

The results show that the parallel section (red circled area) has a negative influence on the neural network's performance.

5.2.4 Model 4 - 8 Stations, One Fanout

The fourth model had the same structure as model 3, but with a shorter fanout of 2 km. It was investigated how a neural network performs if the parallel sections are more similar to each other. Further, it was investigated how the performance of the neural network changes when more data for the training process is used. Once, the earth fault simulations along the affected branch took place in 10% relative to the cable length and once with 2.5% steps to generate more data. The grid model was introduced in the subchapter 3.3.4.4.

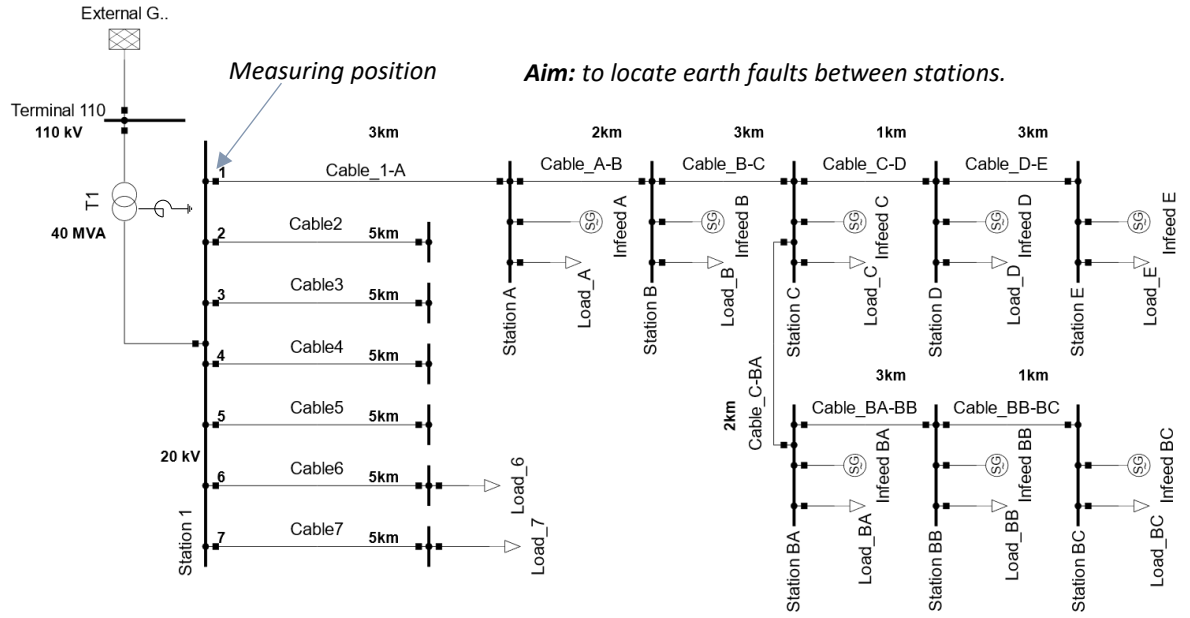
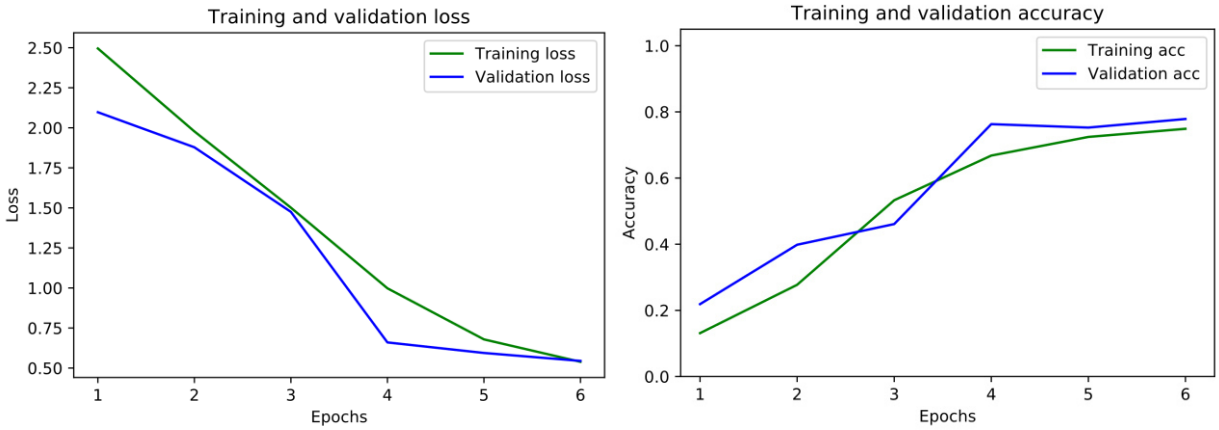


Figure 5-13, earth fault simulation with eight stations on the affected feeder, including a fanout.

5.2.4.1 Results with 10% steps

The training performances in Figure 5-14 show a good generalization of the neural network. Both training and validation loss are declining in parallel, and training and validation accuracy are increasing in the same way. After 6 epochs, a minimum in loss and a maximum of accuracy was reached.



Accuracy of 78%, on the never-before-seen test data set.

Figure 5-14, training and validation loss (left), training and validation accuracy (right)

For the never-before-seen test data set, an accuracy of 78% was achieved. The accuracy of the test set is 10% lower than with the previous grid model.

Figure 5-15 shows the resulting probability distribution. Same as before, in the beginning, the neural network predicted the sections of the earth faults almost always 100% correctly. When it comes to the parallel cable, inaccuracy comes up. The neural network can hardly detect the right location. In the end, when one cable is running out, the probability for the right prediction increased again. Different from the

previous grid model, in the red circled area, the neural network was not really able to predict, even with a low probability, the correct earth fault location.

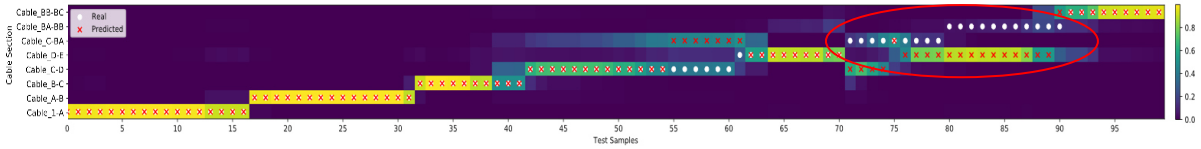
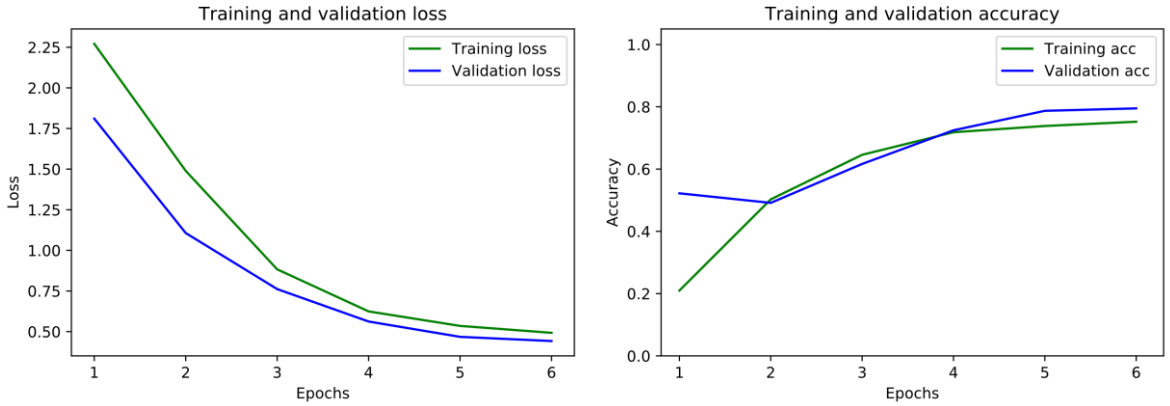


Figure 5-15, diagram for the probability distribution on 100 randomly chosen test samples

5.2.4.2 Results with a smaller step rate of 2.5%

As described above, in this model, a more in-depth look was taken at what impact a lower step rate of 2.5% has on the training and test results.

Although the training results almost led to the same resulting values as before, the training performances show a better generalization. The loss and accuracy lines are running more parallel and are, therefore, less noise. It is an indicator of well-structured data sets.



Accuracy of 78% on the never-before-seen test data set.

Figure 5-16, training and validation loss (left), training and validation accuracy (right)

Figure 5-17 looks at the first look almost the same as Figure 5-15. On the first three cable sections, the predictions of the neural network on the correct earth fault locations were very high and a bit lower on the transition points. In the parallel section, some differences show up. Although the accuracy on the test data set with 78% was equal, the probability distributions of the predictions on the wrong fault sections are lower than in Figure 5-15 (red circled area). Thus, in case the fault location is predicted incorrectly, with the remaining probability on the other cable sections, it would still be possible to get an idea of where the fault might be located. When one cable was running out at the end of the parallel section, the probability of the correct predicted earth fault location increased again.

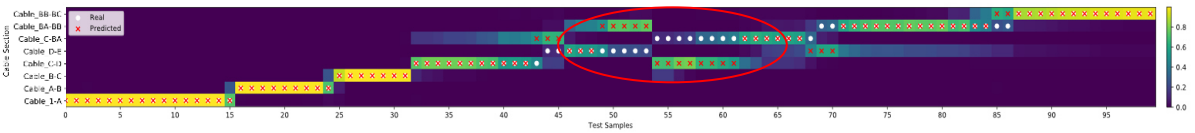


Figure 5-17, diagram for the probability distribution on 100 randomly chosen test samples

5.2.4.3 Conclusion

With more data samples, the training performances showed a slightly better generalization. The visualization of the probabilities also showed a very small positive impact on the test data samples. It was easier to see where the fault is, even if incorrectly predicted. The 10% step rate was used for further investigations in order to be able to compare the various grid models.

5.2.5 Model 5 - 8 stations including one fanout, and without a power infeed in station C

In this model, it was examined what influence a single power infeed at station C has in comparison with the grid model in subchapter 5.2.4.1. The only change to the previous grid model is the replacement of the power infeed at the local grid station C.

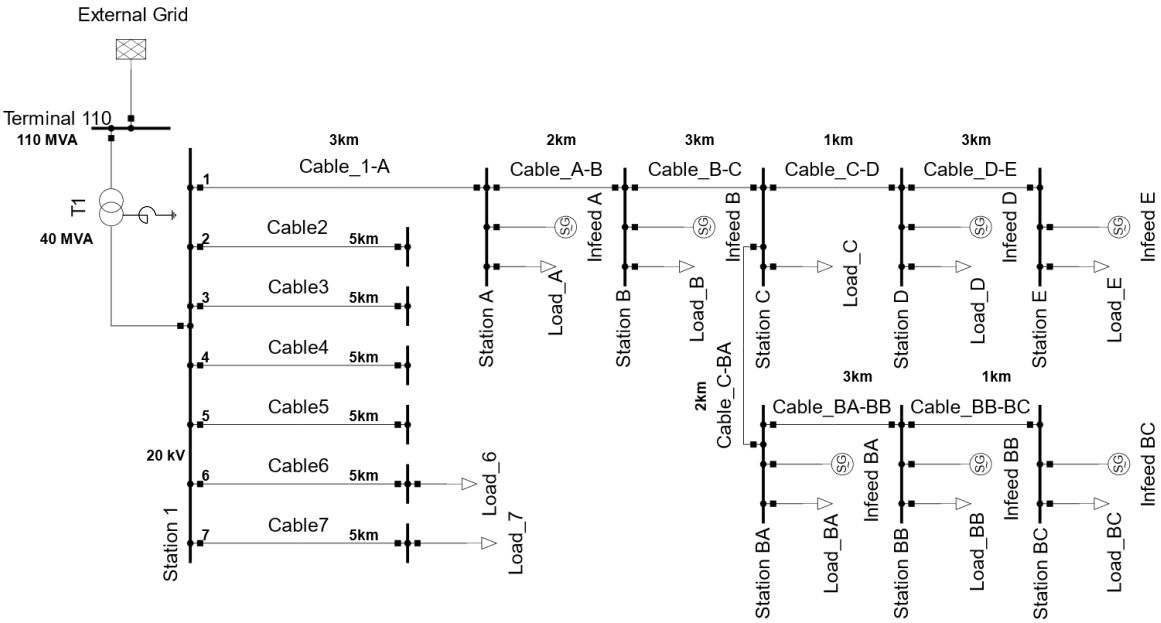
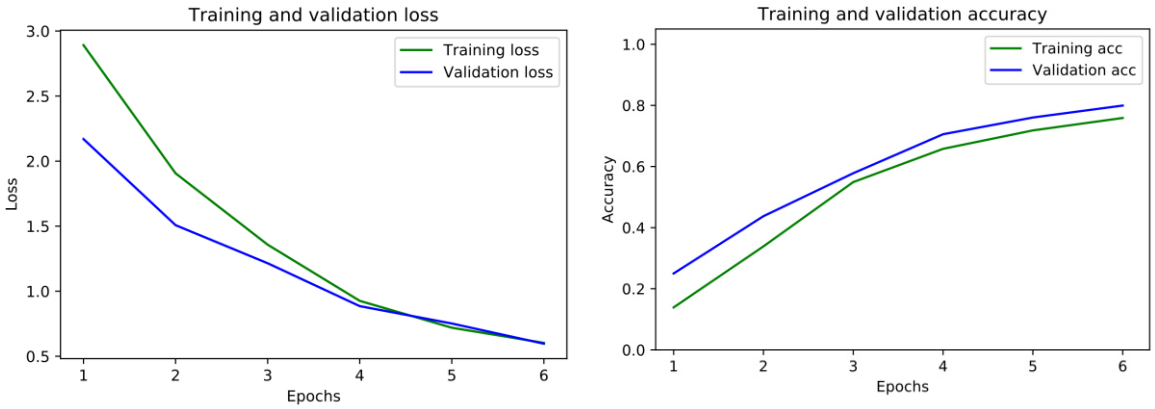


Figure 5-18, earth fault simulation with eight stations on the affected feeder, including a fanout, and without a power infeed in station C

5.2.5.1 Results

The generated 2,560 data samples were split into 70% training data, 15% validation data, and 15% test data. The training performances show a proper generalization of the neural network. The lines are



Accuracy of 78%, on the never-before-seen test data set.

Figure 5-19, training and validation loss (left), training and validation accuracy (right)

running very parallel. The training was stopped after six epochs. On the never-before-seen test data, an accuracy of 78% was reached. That is the same amount of accuracy as before.

The probability visualization in Figure 5-20 shows the same characteristics as the previous one in Figure 5-17. The neural network correctly predicted the earth faults along the first three cables with a high probability. The red circled area looks nearly the same as before. Only the number of test samples differs in this area. In the end, the probability of the correctly predicted earth fault locations increased again.

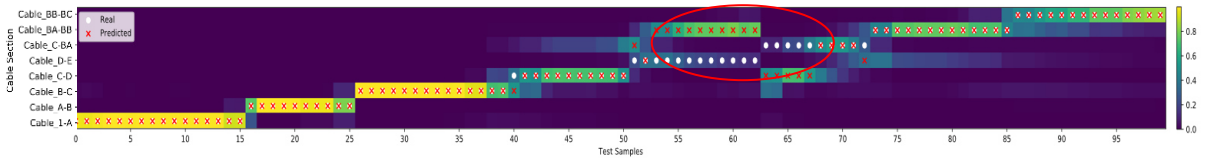


Figure 5-20, diagram for the probability distribution on 100 randomly chosen test samples

The results of the test data set indicate a small influence of the power infeed at the station C, and look very close to the results in Figure 5-17 with the 2.5% steps. The quality of the predictions is a bit higher.

5.2.5.2 Conclusion

The removal of the single power infeed at station C has not changed the results of the neural network, since the results differ only slightly from each other, compared to the previous results in subchapter 5.2.4.1.

5.2.6 Model 6 - 8 Stations, one fanout, and no power infeeds

In this subchapter, it was examined how the neural network's behavior changes if no power infeeds at the affected branch are included, whereas the rest of the grid parameters remained the same.

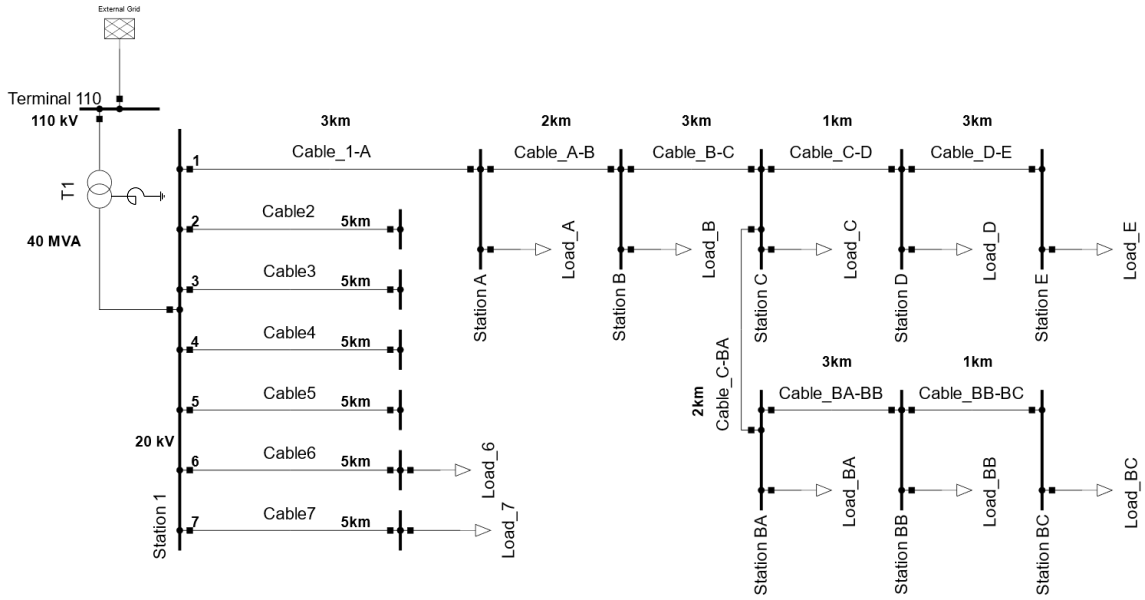
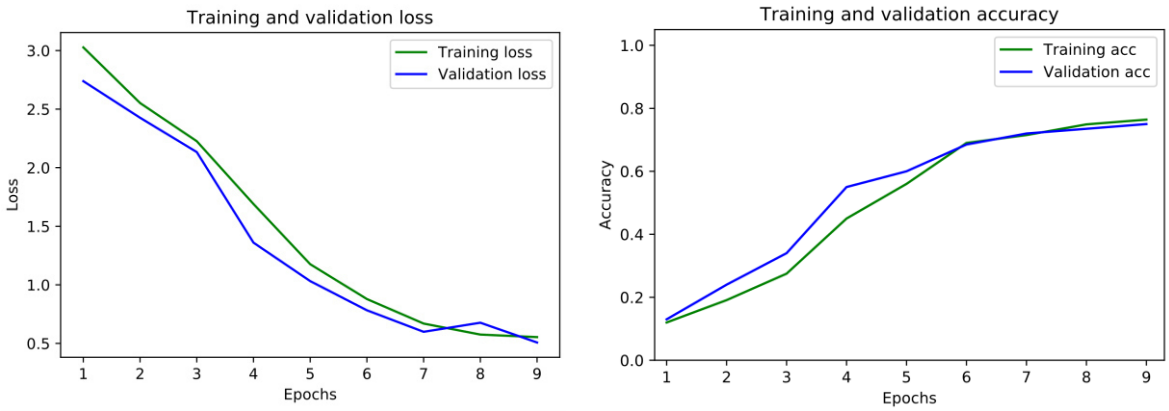


Figure 5-21, earth fault simulation with eight stations on the affected feeder, including a fanout, and no power infeeds

5.2.6.1 Results

The training performances show a good generalization (Figure 5-22). The training was stopped after nine epochs. On the never-before-seen test dataset, an accuracy of 78% was reached. This is very close to the accuracy of the validation data at the end of the training process.



Accuracy of 78%, on the never-before-seen test data set.

Figure 5-22, training and validation loss (left), training and validation accuracy (right)

Figure 5-23 shows the probability distributions of the neural network's predictions. In comparison to Figure 5-20, incorrect predictions are predicted with a slightly lower probability.

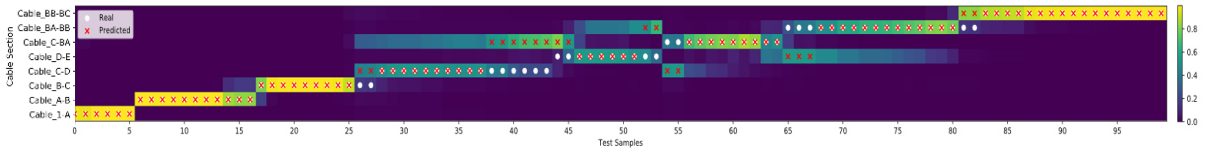


Figure 5-23, diagram for the probability distribution on 100 randomly chosen test samples

5.2.6.2 Conclusion

Power infeeds have a negative impact on the performance of earth-fault-locating neural networks.

5.2.7 Model 7 - 8 Station, and open circuit operation

Next, it was examined how a neural network reacts if the affected branch is in open circuit operation, to find out what influences the loads and power infeeds have. In comparison to the grid model in subchapter 5.2.4, all loads, and power infeeds were removed from the faulty branch. The simulation took place in the same way as before. The earth faults were simulated along the cables in 10% steps.

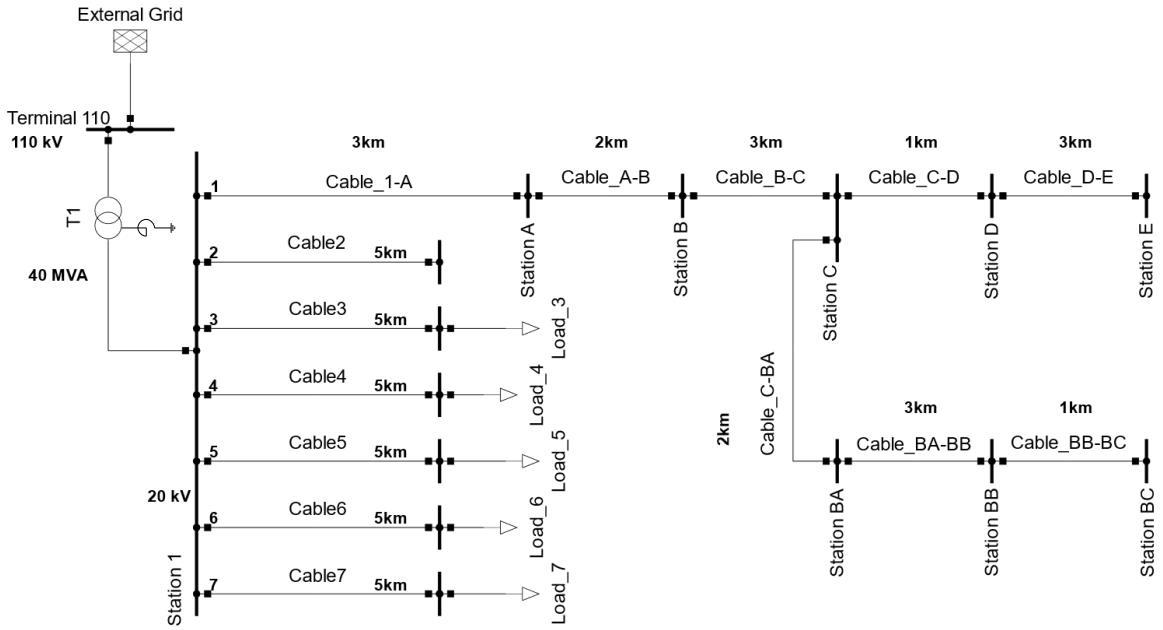
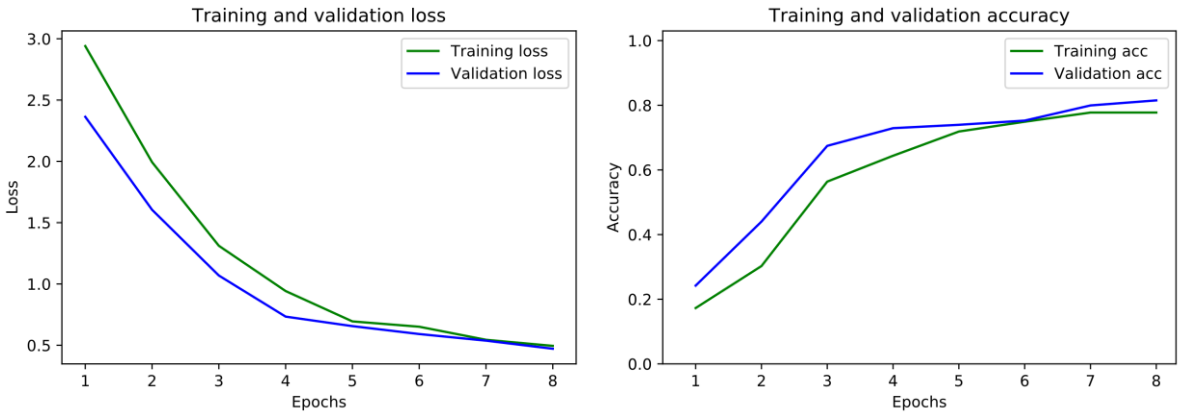


Figure 5-24, Earth fault simulation with eight stations on the affected branch, including a fanout and open circuit

5.2.7.1 Results

The training performances indicate a good generalization. As before, the lines are increasing and decreasing in parallel. On the never-seen-before test data set, an accuracy of 81% was reached. That is about 3% higher than was reached with the grid model in 5.2.4.1.



Accuracy of 81%, on the never-before-seen test data set.

Figure 5-25, training and validation loss (left), training and validation accuracy (right)

Figure 5-26 shows the probability distribution. In the diagram, it can be seen that the probabilities are more evenly distributed between the parallel cables, than in subchapter 5.2.4.1. The main difference is that the incorrectly predicted fault locations are predicted with a lower probability.

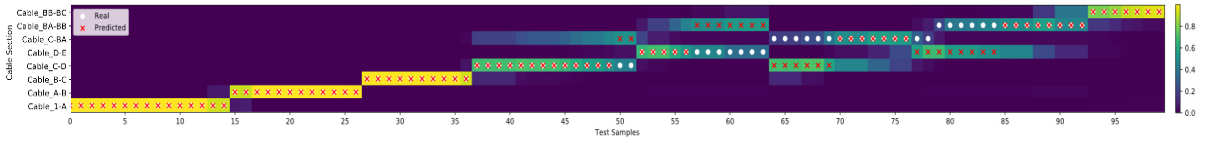


Figure 5-26, diagram for the probability distribution on 100 randomly chosen test samples

5.2.7.2 Conclusion

In general, the results indicate that loads and decentralized generation have a negative impact on earth-fault-locating neural networks in parallel sections.

5.2.8 Model 8 - 7 stations with an identical parallel section

The last exemplary model consists of 7 stations along the affected branch. The stations of the parallel section are totally identical, including the same lengths of the cable connections. It also consists of the same characteristics for all local grid stations. The emphasis lies in this grid model on the identical parallel section.

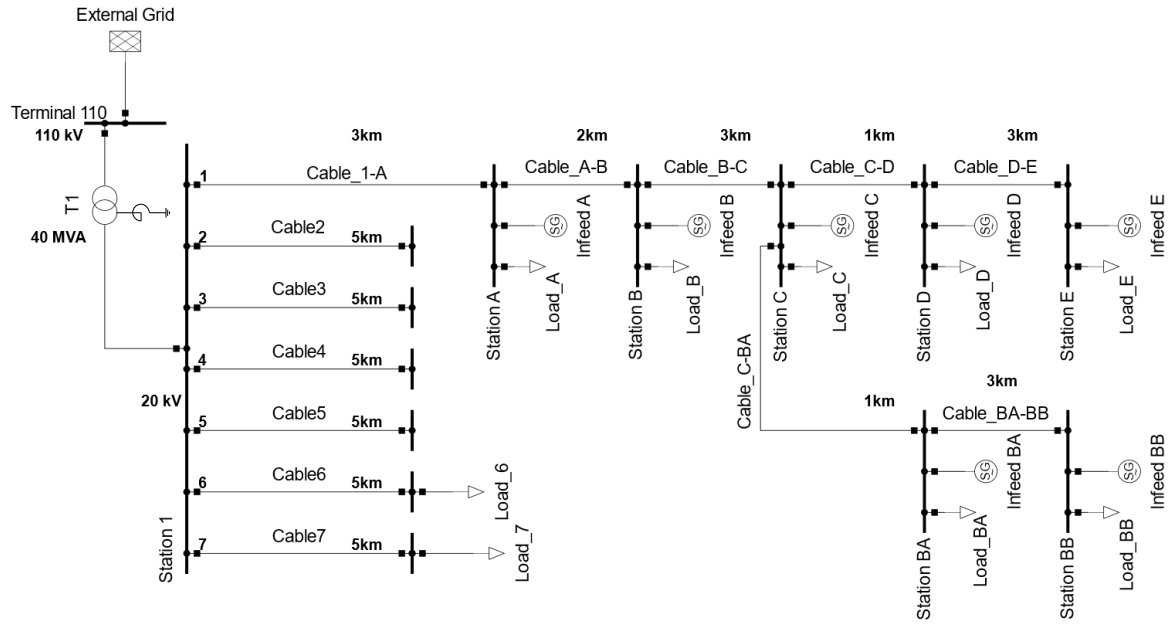
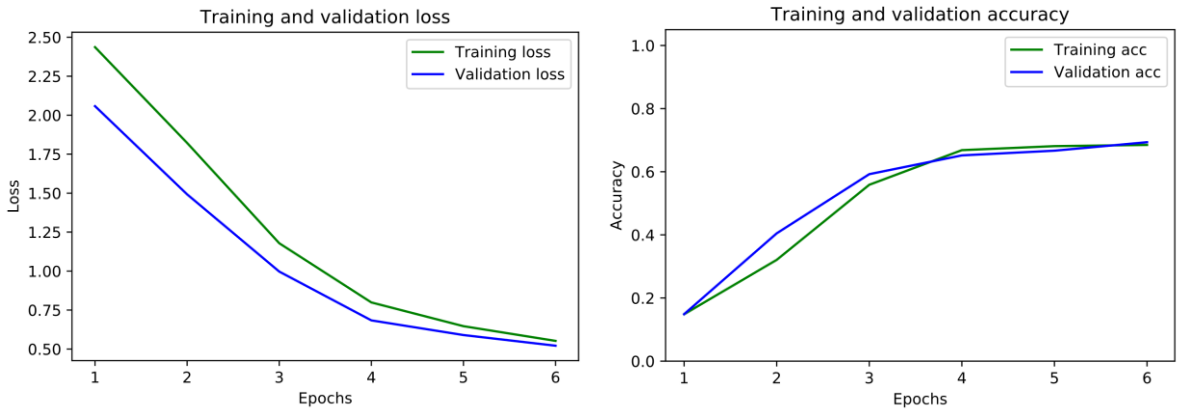


Figure 5-27, earth fault simulation with seven stations on the affected feeder, with a total identically parallel section

5.2.8.1 Results

In Figure 5-28, in the left diagram, the training and validation loss is declining in parallel—indicative of a good generalization. The training stopped after 6 epochs at an optimum, so the neural network is not overfitting. On the right side, the training and validation accuracy is vice versa the loss diagram, rising in parallel. For the never-before-seen test data set, an accuracy of 73% was reached.



Accuracy of 73%, on the never-before-seen test data set.

Figure 5-28, training and validation loss (left), training and validation accuracy (right)

Figure 5-29 shows the probability distribution of 100 randomly chosen test data samples. In the beginning, all earth faults were predicted correctly, which was the same for the second and third cables. On the transition points from one cable to another, the probability for the correct prediction decreased a little. When it comes to the parallel section, the neural network was not able to predict correctly. The probability of a correct prediction in this area is almost always around 50 to 50%.

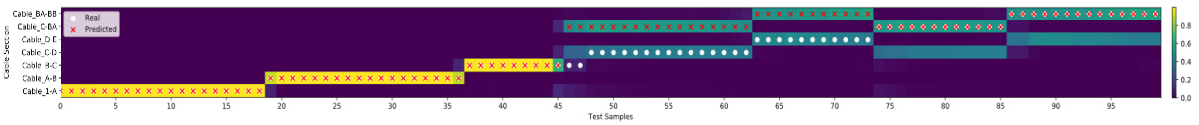


Figure 5-29, diagram for the probability distribution on 100 randomly chosen test samples

5.2.8.2 Conclusion

The neural network had trouble predicting the earth-faulted sections. It was not possible for the neural network to identify the earth-faulted cable in the parallel section. This shows that the main inaccuracy and challenge for the earth-fault-locating neural network is coming from the parallel section.

6 Application of Earth-Fault-Locating Neural Networks

In this chapter, a possible application of earth-locating neural networks is introduced. It is described what kind of training data is required and how such neural networks can be implemented in operating systems for distribution grids.

6.1.1 Required Data

The data used for developing neural networks plays an essential role. Neural networks are only as good as the data on which they are trained. If the task is to classify earth-faulted sections of different branches with different topologies, it is important to represent all types of earth faults and grid parameters in the training data. This includes:

- Type of cables and/or overhead lines
- Length of cables and/or overhead lines
- Earth fault position
- Kind of earth fault, including different transition impedances
- The topology of the grid
- The number of local grid stations with possible power
- Line-to-ground capacities of the whole grid
- Parameters of the arc-suppression coil, including compensation parameters.

With the known parameters of the grid, it is possible to simulate earth faults in software like DIgSILENT PowerFactory, while varying all parameters which might influence earth faults in real operating systems. Through the parameter variation, large amounts of already labeled data are generated. The more that earth fault influencing possibilities are simulated, the higher the chance that the trained neural network can classify earth-faulted sections. For each branch, such training data is required. If an earth fault occurs, the parameters of which are not represented in the training data, the neural network still outputs some results. However, these results will mostly contain misleading information. In other words, the output will be useless.

6.1.2 Earth-Fault-Locating Neural Network

The inputs for an earth-fault-locating neural network are the voltages and currents measured in the substation at the outgoing branch. In Figure 6-1, a possible application is shown. The inputs are measured at the outgoing branch (red circled area). If an earth fault is detected, the neural network outputs the predicted section, including the possibility distribution for all sections. In a best-case scenario, the neural network would predict the correct section.

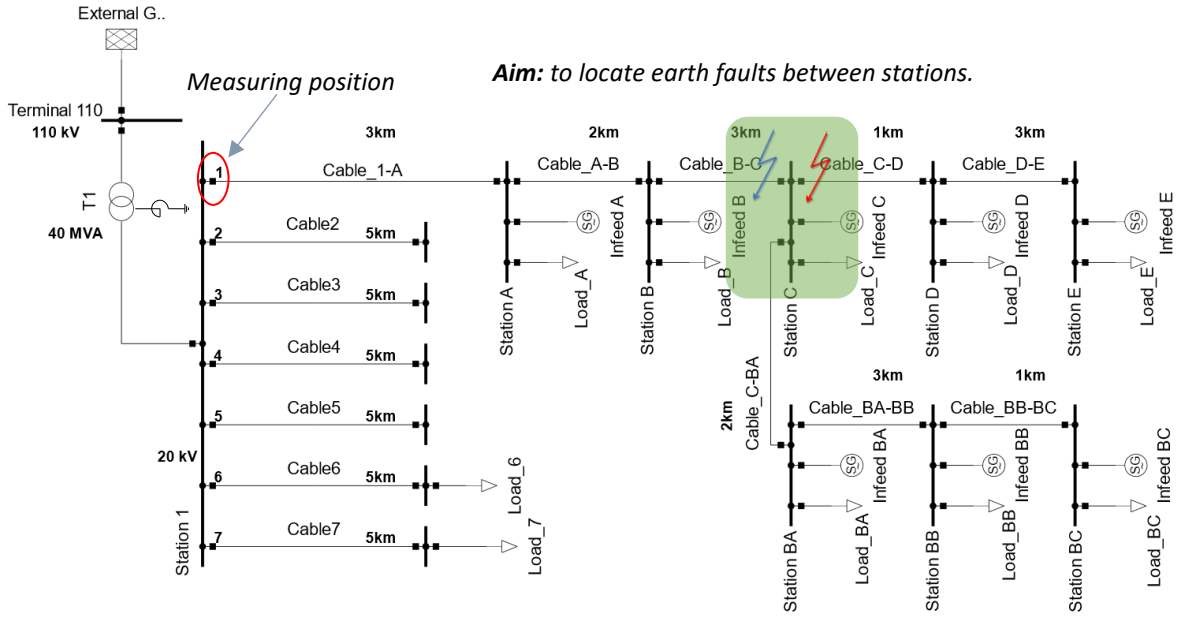


Figure 6-1, Earth fault locating neural network

If the neural network predicted the wrong section through the possibility distribution, the area of fault could still be narrowed. In Figure 6-1, the neural network predicted the wrong section, marked in blue, between station B and C. The actual fault is located between station C and D.

By the visualization of the probability distribution, it is possible to see where the area of fault is. The incorrectly predicted section got a probability of 40%. In contrast, the actual faulted section was predicted with 38%, just 2% less, as shown in Figure 6-2. The section between station C and BA got the remaining percent. With this information, a narrowed area of fault can become visible as the green area in Figure 6-1. In this scenario, all other sections are negligible because of their low probabilities.

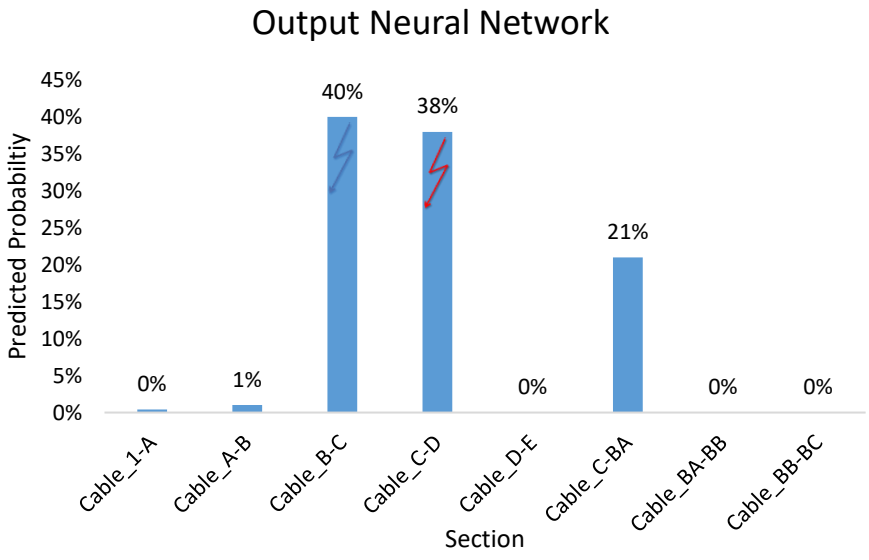


Figure 6-2, Output Neural Network

6.1.3 Application in operating medium voltage networks

Since neural networks output predictions every time they are fed with input data, it has to be ensured that this input data contains the correct information. In other words, for an earth-fault-locating neural network, the input data needs to represent an earth fault. Otherwise, the neural network will probably predict some sections but without any reliability.
It is required that earth-fault-locating neural networks are only fed with real earth faults. Otherwise, it would take a lot of effort to simulate and further represent all kinds of other possibilities in the training data. Here is where the classical earth-fault-detecting methods come in. Neural networks can be combined with classical earth-fault-detecting methods to increase the accuracy of such applications. In the following, a possible combination is introduced.

6.1.3.1 Combined with the transient method

A combination with the transient method is examined. The transient method with the qu-algorithm has the advantage to detect earth faults even with high transition impedances of a few kΩ. It can detect the earth-faulted branch with high accuracy but not locate the earth-faulted section. If an earth fault is detected, the measured data can be sent into the neural network to predict the earth-faulted section. The measured data can be stored by using a FIFO (first-in, first-out) method with the right size for the neural network. When an earth fault is detected by the transient method, the stored data can be immediately fed in the neural network to predict the earth fault location.
Since common earth-fault relays only measure the displacement voltage and the zero-sequence-current of the monitored branch, additional inputs are required. The three phase-to-earth voltages and the three phase currents are needed as input. Below, the advantages and disadvantages of such a neural network implemented in an earth-fault relay are listed.

Advantages:

- Neural networks combined with proven approaches lower the risk of misleading results from the neural networks.
- The neural network does not have to be trained on the earth-fault detection itself and other possibilities.
- Reduces troubleshooting time
- Outputs the earth-faulted sections immediately

Disadvantages:

- A high wiring effort since only the displacement voltage and the zero-sequence-current is measured in earth-fault-detecting relays
- Exact parameters of the grid need to be known for the data generation
- External visualization of the neural network’s results are required

7 Conclusion

In the beginning, a neural network for fault classification was trained. The performances of the training process and the results of the test data show that the neural network was able to correctly classify different types of faults. All simulated faults were correctly predicted with a certainty of at least seventy-five percent.

After the classification task succeeded, the emphasis was on the earth-fault-locating task in compensated medium voltage cable networks, where grid models, which differed in complexity, were examined.

First, earth faults were simulated along one cable section that consisted of four local grid stations, each connected with a 5 km cable. The results pointed out that a neural network can locate with very high accuracy earth faults between four stations connected with the same cable length.

Almost the same results were achieved with a grid having nine stations connected with cables of different lengths. The earth-faulted cable sections were predicted with high accuracy, but the accuracy decreased slightly when the sections were further away from the substation.

In the next steps, the limits of the earth-fault-locating with neural networks were investigated through more complex grid models. In subchapter 5.2.3, it was examined how a neural network performs when it comes to a parallel section. The grid model included one long fanout of 10 km. The results indicated that the neural network had trouble correctly predicting the fault when it occurred in the parallel section. But even when the earth fault location was predicted incorrectly, the neural network was not far away from the actual earth fault section.

In subchapter 5.2.4, the fanout was reduced from 10 km to 2 km in order to see how a shorter fanout in the parallel section influences an earth-fault-locating neural network. In the parallel section, the results showed that the neural network was unable to identify the correct earth fault position at some points. On the same grid, four times more data was generated through a lower step rate of 2.5% of the simulated earth fault positions to investigate the influence of more samples. As a result, the accuracy of the test samples remained at 78%, but the visualization of the probability distribution indicated a slightly better quality. The probabilities were distributed in a way that made it easier to find out where the earth fault might be located.

Using the same grid model as before, the impact of loads and power infeeds on the performance of earth-fault-locating neural networks was examined. In the first scenario, the single power infeed at the cable fanout station was removed. In the second, all power infeeds at the affected branch were removed. In the third scenario, all loads and power infeeds were removed to see what happens if the faulty branch is in open circuit operation.

When the decentralized generation at the cable fanout station was removed, the results showed only very slight improvements. The accuracy of the test data set remained at 78%. The differences were in the probability distributions. Incorrect predictions were detected with a slightly lower probability. But in

sum, it was not possible to conclude that the removal of a single power infeed had a significant positive influence.

In the second scenario, in which all power infeeds were removed, the probability distribution for correct and incorrect prediction improved slightly. However, the accuracy of the never-before-seen test data samples remained the same. Therefore, it was easier to find out where the earth fault might be located even if the prediction was incorrect.

In open-circuit operation, the accuracy of the test data set increased by 3% to 81%, as did the quality of the neural network's predictions. In the parallel section, the probability outputs were more evenly distributed between the parallel cables than before. Therefore, it was easier to get an idea where the fault might be located, even if it was predicted incorrectly. This shows that loads and decentralized generation have a negative impact on earth fault localization in parallel sections.

In the last grid model, a grid model was simulated with a total identical parallel section to find out the impacts of a parallel section itself. In the identical parallel section, the neural network was only able to predict the correct earth fault location with a probability of around 50%. This demonstrated that most inaccuracies came from parallel running cables and showed that the more identical parallel sections are, the higher the inaccuracy of the predictions.

In summary, these results show that neural networks can be very useful tools in the field of network protection. The results depend very much on the grid topologies. The less similar single sections are to each other, the higher the accuracy. The accuracy of the test data sets was at least in the range of 78 to 100%. Almost all predictions with a probability greater than around 80% were predicted correctly. In cases where the earth-faulted sections were predicted incorrectly through the visualization of the probability distributions, it was still possible to indicate where the earth fault might be located. Combined with the transient method for earth fault detection, an earth-fault-locating neural network can help to identify an earth-faulted section immediately, which reduces troubleshooting times.

In general, neural networks are only as good as their training data. If the input is different than the represented cases in the training data, the neural network will output a result, but the result will not include any meaningful information. In this master's thesis, the neural networks were only trained on selected faults. For a real-world application, much more data, with all possible types of faults and parameters, would be needed. A combination of earth-fault-locating neural networks with already-proven earth-fault-detection methods, like the transient method, can help to improve the reliability of such earth-fault-locating neural networks, as shown in chapter 0.

8 References

[1] K. F. Schäfer, *Netzberechnung, Verfahren zur Berechnung elektrischer Energieversorgungsnetze*. Springer Vieweg, Wiesbaden, 2019.

[2] G. Druml, “Innovative Methoden zur Erdschlussortung und Petersen-Spulen Regelung,” Technische Universität Graz, 2012.

[3] D. Oeding and B. R. Oswald, *Elektrische Kraftwerke und Netze*. Springer Vieweg, Wiesbaden, 1978.

[4] L. Ficker, “Schutz und Versorgungssicherheit elektrischer Systeme,” 2008.

[5] A. Muth, “Die Sternpunktbehandlung,” 2020. [Online]. Available: https://www.schutztechnik.com/posts/die-10-wichtigsten-fehlerarten-im-drehstromnetz.

[6] V. Crastan, *Elektrische Energieversorgung 1*. Springer Vieweg, Wiesbaden, 2000.

[7] IEAN TU Graz, “Schutz und Versorgungs- sicherheit elektrischer Energiesysteme - LU,” 2019.

[8] K. Heuck, K.-D. Dettmann, and D. Schulz, *Elektrische Energieversorgung*. Springer Vieweg, Wiesbaden, 2013.

[9] E. Fuchs, “Kritische Betrachtung des Löschverhaltens in kompensierten 20-kV-Netzen Kritische Betrachtung des Löschverhaltens in kompensierten 20-kV-Netzen,” Graz University of Technology, 2013.

[10] E. Hufnagl, “Choice of system neutral treatment and earth fault protection in aged medium voltage cable networks,” Graz University of Technology, 2018.

[11] A. J. Schwab, *Elektroenergiesysteme*. Springer Vieweg, Berlin, Heidelberg, 2020.

[12] G. Druml, “Info - Brief Nr . 18 Wischerrelais für hochohmige , intermittierende und wiederzündende Erdschlüsse (Teil 2),” *A. Eberle GmbH Co. KG*, no. 18, pp. 1–2, 2013.

[13] G. Druml, “Info - Brief Nr . 15 Erdschlussortung nach dem Oberschwingungsverfahren (Teil 2),” *A. Eberle GmbH Co. KG*, no. 15, pp. 2–3, 2013.

[14] G. Druml, “Info - Brief Nr . 14 Erdschlussortung nach dem Oberschwingungsverfahren (Teil 1),” *A. Eberle GmbH Co. KG*, no. 14, pp. 2–3, 2013.

[15] DIgSILENT-GmbH, “DIgSILENT PowerFactory.” 2020.

[16] N. Essl and H. Renner, “Low-Voltage-Ride-Through-Assistenzsystem für dezentrale Erzeugungseinheiten,” *Elektrotechnik und Informationstechnik*, vol. 131, no. 8, pp. 309–315, 2014.

[17] S. J. Russell and P. Norvig, *Artificial Intelligence A Modern Approach*, vol. 3. New Jersey: Pearson Education, Inc., 1995.

[18] F. Challet, *Deep Learing with Python*. New York: Manning Publications Co, 2018.

[19] A. Agarwal *et al.*, “TensorFlow : Large-Scale Machine Learning on Heterogeneous Distributed Systems,” 2015.

[20] Mazur Matt, “A Step by Step Backpropagation Example,” 2015. [Online]. Available: https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/.

9 Appendix

9.1 Program neural network for classification

This is a code example showing a neural network for an earth-fault-locating task.

The code is structured in the following way:

-Data preparation

-Neural network

-Analyzing results

The Keras library from Google is used with the framework TensorFlow.

Import of needed libraries for reading files, data handling, and plotting.

```
import pandas as pd
import numpy as np
import os
import random
import glob
from os.path import normpath, basename
import matplotlib.pyplot as plt
```

Data preparation

All CSV files are read out at the storage location. In the beginning, all files need to be randomly shuffled to avoid any misleading structures in the training process later on. In this, 2,560 files are loaded and randomly shuffled.

```
path = r'C:\Users\Matthäus\Documents\02-Masterarbeit\06-DeepLearning\Sim_8 Stati
ons_ohne Einspeisung'
all_files = glob.glob(path + "/*.csv")

random.shuffle(all_files)
print(len(all_files))
```

```
2560
```

The data files are split into 15% for the validating task and 15% for testing, and the remaining 70% is used for training.

```
validation_data_files = all_files[:384]
test_data_files = all_files[384:768]
training_data_files = all_files[768:]
```

```
training_data_files[8]

'C:\\Users\\Matthäus\\Documents\\02-Masterarbeit\\06-DeepLearning\\Sim_8 Station
s_ohne Einspeisung\\2310000030335312.csv'
```

The first two characters of each filename identify the fault and the earth-faulted section of the branch. For example, 24 stands for an earth fault in phase 2 in the fourth section of the branch.

Since samples are labeled based on the names of their files, the labels and their contained data need to be separated into arrays. In the case of the training data files, the actual data is stored in the array traning_data, and the associated respective labels are stored in training_label.

This is done using a for loop. The filenames are read out (line 11), as well as the data (lines 15 to 18), including some form changes. In the end of the for loop, the out readings are stacked above to each other.

```
#Training Data
training_data = []
df = []
training_label = []
number = []

for filename in training_data_files:
    #label
    name = np.array(os.path.basename(filename))

    number = name.astype('<U2')
    number = number.astype('int32')

    #features
    df = pd.read_csv(filename, sep=";",decimal=".")    #get files
    df = df.apply(lambda x: x.str.replace(',','.')) #replace comma by dot
    df = np.array(df)   #convert in numpy
    feature_voltage = df[1:,1:7]  #get voltages and currents

    df= np.vstack([feature_voltage])
    df = df.astype('float32')

    training_data.append(df)
    training_label.append(number)
```

The same is done for validation data and test data, as follows.

```
validation_data = []
df1 = []
validation_label = []
number1 = []

for filename1 in validation_data_files:

    df1 = pd.read_csv(filename1, sep=";",decimal=".")    #get files
    df1 = df1.apply(lambda x: x.str.replace(',','.')) #replace comma by dot
    df1 = np.array(df1)   #convert in numpy
    feature_voltage = df1[1:,1:7]  #get voltages and currents

    df1= np.vstack([feature_voltage])
    df1 = df1.astype('float32')
    #label
    name1 = np.array(os.path.basename(filename1))
    number1 = name1.astype('<U2')
    number1 = number1.astype('int32')

    validation_data.append(df1)
    validation_label.append(number1)
```

The test data does not need to be shuffled and can be sorted to make analyzations of the neural network easier later on.

```
test_data_files = sorted(test_data_files)

test_data = []
df2 = []
test_label = []
number2 = []

for filename2 in test_data_files:

    df2 = pd.read_csv(filename2, sep=";",decimal=".")    #get files
    df2 = df2.apply(lambda x: x.str.replace(',','.')) #replace comma by dot
    df2 = np.array(df2)   #convert in numpy
    feature_voltage = df2[1:,1:7]  #get voltages and currents

    df2= np.vstack([feature_voltage])
    df2 = df2.astype('float32')
    #label
    name2 = np.array(os.path.basename(filename2))
    number2 = name2.astype('<U2')
    number2 = number2.astype('int32')

    test_data.append(df2)
    test_label.append(number2)
```

Next, the data and label sets are modified so that the data sets have the shape that is later needed for feeding the neural network.

```
training_data = np.reshape(training_data, ((len(training_data)), 4002,6,1))
validation_data = np.reshape(validation_data, ((len(validation_data)), 4002,6,1)
)
test_data = np.reshape(test_data, ((len(test_data)), 4002,6,1))
```

```
training_label = np.reshape(training_label, len(training_label))
validation_label = np.reshape(validation_label, len(validation_label))
test_label = np.reshape(test_label, len(test_label))
```

Import of needed libraries for building a neural network.

```
from tensorflow.keras import models
from tensorflow.keras import layers
from keras.utils import to_categorical
from keras import regularizers
```

In the following, the labels are categorically encoded by one-hot encoding. Each label is embedded in an all-zero vector but with a one in the label index.

```
train_labels = to_categorical(training_label)
validation_labels = to_categorical(validation_label)
test_labels = to_categorical(test_label)
```

Neural network

Now, the data preparation is complete, and the data is ready to be fed into a neural network.

Since there is no rule regarding how the architecture of a neural network for specific data must look, it was discovered after many repetitions that the following model works well on this data.

The layers are literally chained together. On the bottom layer (line 2), the input shape of the training data is required. In this case, one sample has a shape of (4002,6,1).

The following layer automatically takes as input shape the output shape of the previous layer.

The chosen architecture is a mix of convolutional and densely connected layers.

Convolutional layers (also called convnets) as Conv2D take as input 3D shapes; that is why the data was reshaped in a previous step. The advantage of convnets is that once they have learned a specific pattern, that pattern can be recognized even in other places and in other samples. The learned patterns are invariant, which makes convnets very efficient in computer vision tasks.

Softmax is used as last-layer activation. Softmax outputs a probability distribution over all classes. The class with the highest probability is the neural network's choice for specific input.

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(4002,6,1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(32, (3, 2), activation='relu'))
model.add(layers.MaxPooling2D((2, 1)))
model.add(layers.Conv2D(64, (3, 1), activation='relu'))

model.add(layers.Flatten())
model.add(layers.Dropout(0.3))
model.add(layers.Dense(32, kernel_regularizer=regularizers.l2(0.002), activation
='relu',input_shape=(None, 4002, 6)))
model.add(layers.Dense(29, activation='softmax'))
```

With this model architecture, the neural network consists of 2,055,549 parameters in total.

```
model.summary()

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 4000, 4, 32)       320
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 2000, 2, 32)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 1998, 1, 32)       6176
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 999, 1, 32)        0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 997, 1, 64)        6208
_________________________________________________________________
flatten_1 (Flatten)          (None, 63808)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 63808)             0
_________________________________________________________________
dense_2 (Dense)              (None, 32)                2041888
_________________________________________________________________
dense_3 (Dense)              (None, 29)                957
=================================================================
Total params: 2,055,549
Trainable params: 2,055,549
Non-trainable params: 0
_________________________________________________________________
```

The compilation step defines three things:

Optimizer: How the network updates itself while training. RMSprop is the most famous optimizing algorithm in deep learning.

Loss: How the neural network measures its performance while training is defined in the loss function; 'categorical_crossentropy.'

Metrics: Allows one to monitor the accuracy of the neural network during the training process.

```
model.compile(optimizer ='rmsprop',
              loss = 'categorical_crossentropy',
              metrics=['accuracy'])
```

Once the data preparation is complete and the architecture of the neural network has been built, training can begin.

The training starts by calling the fit() method. It returns as object the history of the training process. Inputs are the training data with their corresponding labels (targets). In this case, the whole training data goes five times (5 epochs) through the training process in batches of 50 samples. After each epoch, the neural network is evaluated by the validation data set.

```
history = model.fit(training_data,
                    train_labels,
                    epochs=5,
                    batch_size=50,
                    validation_data =(validation_data,validation_labels))
```

```
Train on 1792 samples, validate on 384 samples
Epoch 1/5
1792/1792 [==============================] - 6s 3ms/sample - loss: 2.7179 - acc:
0.1373 - val_loss: 2.1854 - val_acc: 0.2500
Epoch 2/5
1792/1792 [==============================] - 4s 2ms/sample - loss: 1.6159 - acc:
0.4554 - val_loss: 1.2003 - val_acc: 0.6198
Epoch 3/5
1792/1792 [==============================] - 4s 2ms/sample - loss: 0.9006 - acc:
0.6836 - val_loss: 0.7853 - val_acc: 0.6458
Epoch 4/5
1792/1792 [==============================] - 4s 2ms/sample - loss: 0.6498 - acc:
0.7260 - val_loss: 0.5874 - val_acc: 0.6927
Epoch 5/5
1792/1792 [==============================] - 4s 2ms/sample - loss: 0.5535 - acc:
0.7573 - val_loss: 0.5158 - val_acc: 0.7786
```

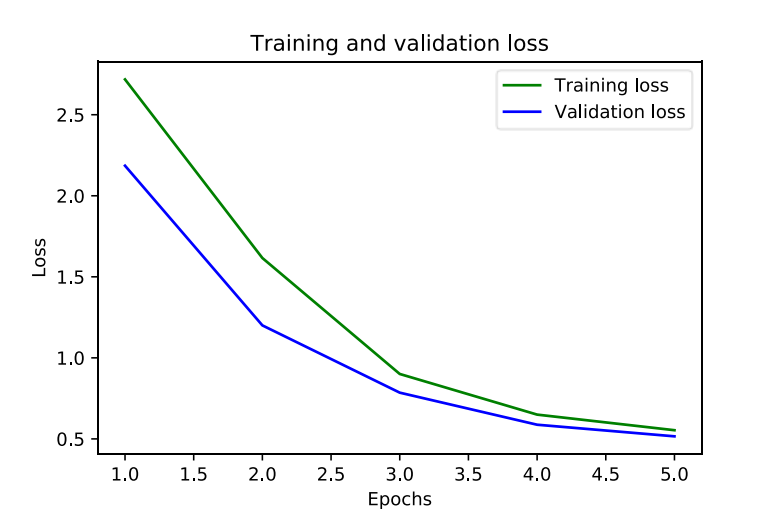
Analyzing results

The history object contains four entries: training loss, validation loss, training accuracy, and validations accuracy.

Below, the training and validation loss are plotted side by side, as well as the training and validation accuracy.

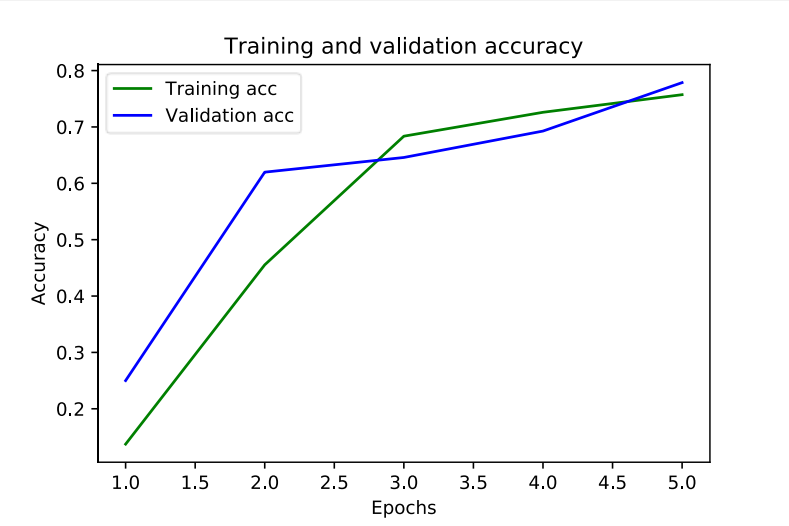
```
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1,len(loss)+1)

plt.plot(epochs, loss,'g',label = 'Training loss')
plt.plot(epochs, val_loss, 'b', label = 'Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.savefig('Training-Validation-loss.svg')
plt.show()
```




```
acc = history.history['acc']
val_acc = history.history['val_acc']

plt.plot(epochs, acc,'g',label = 'Training acc')
plt.plot(epochs, val_acc, 'b', label = 'Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.savefig('Training-Validation-accuracy.svg')
plt.show()
```



With the following two plots, the training of the neural network is evaluated. At this point, it can be seen whether the architecture of the neural network needs to be modified (for example, adding layers) and be trained again, depending on underfitting or overfitting.

Once the training results indicate a proper generalization, it is finally evaluated through the test data set, in which data was not seen by the neural network before. This is done as follows.

```
model.evaluate(test_data,test_labels)
384/384 [==============================] - 0s 903us/sample - loss: 0.5219 - acc: 0.7604
[0.5219109567503134, 0.7604167]
```

It is always recommended to save the neural network

```
model.save('name of neural network')
```

With predict(), all prediction from the test data can get stored in an array.

```
predictions = model.predict(test_data)
```

With argmax(), the class with the highest probability distribution can be displayed. For example, for the sample 100.

```
np.argmax(predictions[100])
23
```

To receive the single distributions over the classes of a specific sample:

```
for i in range(21,len(predictions[0])):
    print(predictions[100][i])
```

```
0.0016532267
0.13189343
0.8635561
0.0017368143
0.0002396272
7.1554955e-06
7.252301e-06
0.00024241534
```

In the next step, a Hovmoller diagram is created to evaluate the samples, including the probability distributions over the classes. Before, only the predicted class with the highest probability was matched with the target. Below, the probability distribution is also shown.

First, classes with the highest probabilities out of the results from the test data set are stored in the array named y_test_label.

```
predictions = model.predict(test_data)
y_test_label =[]
for i in range(len(predictions)):
    y_test = np.argmax(predictions[i])
    y_test_label.append(y_test)
```

Second, all probability distributions are stored in a separate array named z_distribution.

```
z_distribution = []

for j in range(0,384):
    z_test = predictions[j][:29]
    z_distribution.append(z_test)

z_distribution = np.transpose(z_distribution)

plt.figure(figsize=(30,5))
plt.scatter(range(0,384),test_label,marker='o',c='w',label = 'Real')
plt.scatter(range(0,384),y_test_label,marker='x',c='r',label = 'Predicted')
plt.imshow(z_distribution, interpolation = "nearest")

plt.colorbar(plt.imshow(z_distribution, interpolation = "nearest"),fraction=0.00
35, pad=0.01)
plt.xticks(np.arange(0, 100, step=5))
plt.yticks( [21,22,23,24,25,26,27,28],[ r'K_1-A', r'K_A-B', r'K_B-C', r'K_C-D',
r'K_D-E', r'K_C-BA', r'K_BA-BB', r'K_BB-BC'])
plt.axis( [0,344.5,20.5,28.5])
plt.legend()
plt.savefig('Hovmuller diagram.svg')

plt.show()
```

Since the diagram above does not really fit on the screen, a smaller diagram with 100 randomly chosen test samples is displayed below.

```
random.shuffle(test_data_files)
test_data_files1 = test_data_files[0:100]
test_data_files1 = sorted(test_data_files1)
```

```
test_data1 = []
df2 = []
test_label1 = []
number2 = []

for filename2 in test_data_files1:

    df2 = pd.read_csv(filename2, sep=";",decimal=".")    #get files
    df2 = df2.apply(lambda x: x.str.replace(',','.')) #replace comma by dot
    df2 = np.array(df2)   #convert in numpy
    feature_voltage = df2[1:,1:7]  #get voltages and currents

    df2= np.vstack([feature_voltage])
    df2 = df2.astype('float32')
    #label
    name2 = np.array(os.path.basename(filename2))
    number2 = name2.astype('<U2')
    number2 = number2.astype('int32')

    test_data1.append(df2)
    test_label1.append(number2)
```

```
test_data1 = np.reshape(test_data1, ((len(test_data1)), 4002,6,1))
test_data1 = test_data1.astype('float32')
test_label1 = np.reshape(test_label1, len(test_label1))
```

```
predictions1 = model.predict(test_data1)
y_test_label1 =[]
for i in range(len(predictions1)):
    y_test1 = np.argmax(predictions1[i])
    y_test_label1.append(y_test1)
```

```
z_axis1 = []
for j in range(0,100):
    z_test1 = predictions1[j][:29]
    z_axis1.append(z_test1)
z_axis1 = np.transpose(z_axis1)
```

```
plt.figure(figsize=(30,5))
plt.scatter(range(0,100),test_label1,marker='o',c='w',label = 'Real')
plt.scatter(range(0,100),y_test_label1,marker='x',c='r',label = 'Predicted')
plt.imshow(z_axis1, interpolation = "nearest")

plt.colorbar(plt.imshow(z_axis1, interpolation = "nearest"),fraction=0.004, pad=
0.01)
plt.xticks(np.arange(0, 100, step=5))
plt.yticks( [21,22,23,24,25,26,27,28],[ r'K_1-A', r'K_A-B', r'K_B-C', r'K_C-D',
r'K_D-E', r'K_C-BA', r'K_BA-BB', r'K_BB-BC'])
plt.axis( [0,99.5,20.5,28.5])
plt.xlabel('Test Samples')
plt.ylabel('Caple Section')
plt.legend()
plt.savefig('Hovmuller diagram 100 samples.svg')

plt.show()
```