BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF RADIO ELECTRONICS
ÚSTAV RADIOELEKTRONIKY

AI-BASED CLASSIFICATION OF RF SIGNALS

KLASIFIKACE RF SIGNÁLŮ POMOCÍ UMĚLÉ INTELIGENCI

SEMESTRAL THESIS
SEMESTRÁLNÍ PRÁCE

AUTHOR
AUTOR PRÁCE

Bc. Samuel Turák

SUPERVISOR
VEDOUCÍ PRÁCE

doc. Ing. Ladislav Polák, Ph.D.

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Semestrální práce

magisterský navazující studijní program **Elektronika a komunikační technologie**

Ústav radioelektroniky

Student: Bc. Samuel Turák ***ID:*** 221059

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Klasifikace RF signálů pomocí umělé inteligenci

POKYNY PRO VYPRACOVÁNÍ:

MP(K)C-M2E: V teoretické části práce podejte přehled o možnostech klasifikace radiofrekvenčních (RF) signálů pomocí hlubokého, případně strojového učení. Uvažujte různé typy RF signálů (např. idle, data, rušící), vysílaných za různých přenosových podmínek (např. obsazenost RF spektra, únikové kanály, vliv environmentální prostředí). Prostudujte si dostupné online knihovny RF signálů, případně vytvořte vlastní knihovnu (tzv. dataset). Navrhněte postup tréninku vybraných technik učení.

MP(K)C-MSE: V experimentální části práce navrhněte a vytvořte skripty k tréninku a aplikaci vybraných technik učení pro klasifikaci RF signálů. V práci uvažujte reálné signály, charakterizované různými parametry, vyskytujícími se při různých přenosových podmínkách. Získané výsledky přehledně zpracujte a diskutujte.

DOPORUČENÁ LITERATURA:

[1] SHI, Yi, and et al. Deep Learning for RF Signal Classification in Unknown and Dynamic Spectrum Environments. In: 2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN) [online]. IEEE, 2019, s. 1-10. ISBN 978-1-7281-2376-9. Dostupné z: doi:10.1109/DySPAN.2019.8935684.
[2] PIJÁČKOVÁ, Kristýna. Radio Modulation Recognition Networks. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Radio Electronics, 2021, 61 p. Bachelor's Thesis.
[3] GOODFELLOW, Ian and et al. Deep Learning: (Adaptive Computation and Machine Learning Series). The MIT Press, 2016. ISBN 978-0262035613.

Termín zadání: 22.9.2023 ***Termín odevzdání:*** 2.1.2024

Vedoucí práce: doc. Ing. Ladislav Polák, Ph.D.

doc. Ing. Lucie Hudcová, Ph.D.
předseda rady studijního programu

ABSTRACT

This thesis is focused on the deep learning-based radio modulation classification. The convolutional neural network is trained and evaluated on two different datasets, one of which was developed for this research using the MATLAB program. The performed approaches to data preprocessing, model training and model evaluation are implemented in the programming environment Python, with the use of newly emerged libraries such as Scikit-learn, and are described in detail. The results are presented and discussed.

KEYWORDS

Modulation classification, deep learning, neural network, model, Python, MATLAB

ABSTRAKT

Táto práca je zameraná na klasifikáciu rádiových modulácií pomocou hlbokého učenia. V práci je konvolučná neurónová sieť trénovaná a hodnotená na dvoch rôznych dátových sadách, pričom jedna z nich bola vyvinutá pre tento výskum použitím programu MATLAB. Vykonané prístupy k predspracovaniu dát, tréningu modelu a hodnoteniu modelu sú implementované v programovacom prostredí Python, s použitím nových knižníc ako Scikit-learn, a sú detailne popísané. Výsledky sú prezentované a diskutované.

KLÍČOVÁ SLOVA

Klasifikácia modulácií, hlboké učenie, neurónová sieť, model, Python, MATLAB

TURÁK, Samuel. *AI-based classification of RF signals*. Semestral Project. Brno: Brno University of Technology, Faculty of Electrical Engineering and Communication, Department of Radio Electronics, 2023. Advised by doc. Ing. Ladislav Polák, Ph.D.

Author's Declaration

Author:	Bc. Samuel Turák
Author's ID:	221059
Paper type:	Semestral Project
Academic year:	2023/24
Topic:	AI-based classification of RF signals

Brno ....
author's signature*

*The author signs only in the printed version.

ACKNOWLEDGEMENT

I would like to thank the advisor of my thesis, doc. Ing. Ladislav Polák, Ph.D. for his professional approach, excellent communication, willingness to always help and positive encouragement.

Contents

List of Figures

List of Tables

Introduction

The emergence of new technologies in wireless communication has not only revolutionized the field but also introduced a new set of challenges for engineers. These challenges stem from the increased demand for higher data rates, better reliability, and an increase number of users in an already crowded spectrum. In response, unconventional methods in signal processing are being explored. This thesis delves into one such method that promises to be a game-changer – the utilization of Artificial Intelligence (AI), with its remarkable ability to learn from data and make informed decisions, in the wireless communication. [2]

The aim of this diploma thesis is to study and apply various approaches for using AI in the classification of different radio frequency (RF) signals. In the long run, emphasis should be placed on classifying different types of RF signals (idle, data, jamming, etc.) transmitted under diverse channel conditions. In the current progress of the thesis, the functionality of previously explored concepts is tested, specifically modulation classification with inconsistent impairments. This approach was discussed in other works such as [1] and [3]. Here it will be implemented on other free-source and created data. The implemented solution and the structure of the developed code will serve as a foundation for the subsequent objectives of the thesis.

This thesis is organized as follows. In the first chapter some basic fundamentals of signal processing are discussed, focusing on subjects crucial to understand in the areas of signal processing used in the practical work of this thesis. Next chapter investigates the basics of Machine learning and Deep learning. The aim of this part can be interpreted as a semi-introduction for researchers who want to get into the field of AI, but could also provide clarification for intermediate AI enthusiasts. The final chapter presents conducted research of AI usage in signal processing, more specific in modulation classification. At the chapter's conclusion the taken approach is described and the achieved results are discussed.

1 Radio Communication Systems

Radio communication systems are a tool with whom information is transformed into a signal [4]. This system was firstly proposed by Shannon in [5]. This chapter will provide a broad overview of the system, with the extent of coverage shown in Fig. 1.1. As it is visible in the picture, transmitter (TX), channel and receiver (RX) are the main blocks of a communication system. Those blocks are described in this chapter briefly. More details about the blocks of Fig. 1.1 can be found in [5], [6], and of course in [7].

1.1 Source and Destination

The input of the TX is a source that would for example convert an acoustic signal into an electromagnetic signal in the case of a microphone. If the source or the destination demand an analog signal, analog-to-digital (A/D) converter in TX and digital-to-analog (D/A) converter in RX can be utilized. Without them the system is considered as pure digital [4].

1.2 Source Encoding and Decoding

The next step in communication system is the reduction of redundancy (data that is bigger than the necessary data needed for transmission when accounting the loss in communication channel) and the reduction of irrelevance (non-essential information that does not contribute to the overall understanding and quality of the signal). The efficiency of source coding is higher when the bit rate between the output and the input of the source coder is lower. This process is also called data compression. On the receiving side when decoding, redundancy that was compressed in coder can be supplemented since it has a recurrent pattern, but irrelevance is completely lost without any effect on the re-created signal [4].

1.3 Channel Encoding and Decoding

Opposite to source encoding, in channel encoding redundant bits are inserted to signal data in a controlled way, to minimize effects of distortions occurring inside the channel [4]. This process is executed by subdividing the original message into k-bit long sequential blocks, which are mapped into n-bit blocks [6]. That means chunks of n-bit blocks are created with the length of k-bits of original data, which implies the additional bits for redundancy in each block amount to $n - k$. The

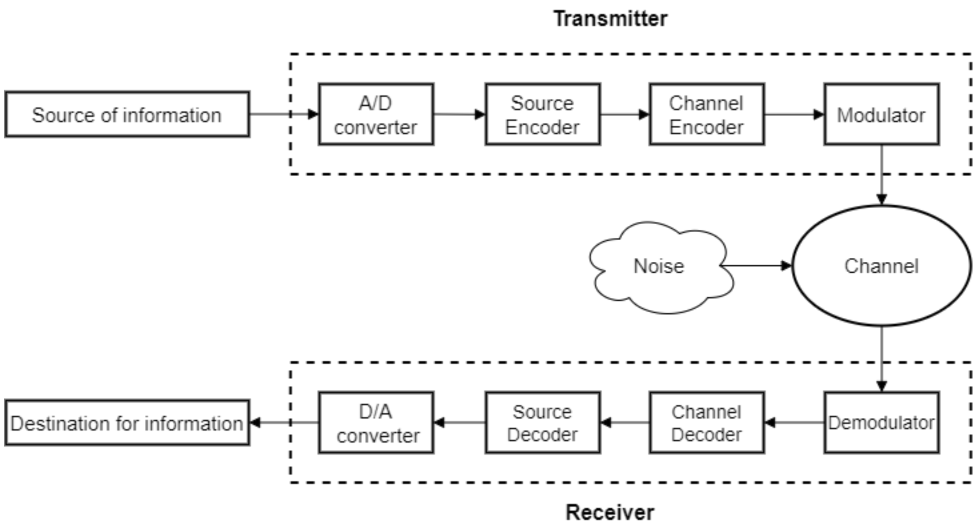


Fig. 1.1: Detailed elements of communication system

quality of the encoder is given by code rate which is equal to

$$r = \frac{k}{n} \tag{1.1}$$

which shows r is always smaller than 1. At the receiving end of the communication system, the decoder with the knowledge of the encoding algorithm and the information encapsulated inside the redundant bits, recreates the original message.

1.4 Modulation and Demodulation

Before the signal is transmitted through channel, the process of modulation needs to be applied. Modulation can be usually described as a process, where some properties of a carrier signal (amplitude, frequency or phase in case of an analog carrier) are changed in accordance with original signal [4]. The transmitted signal will adopt the frequency of the carrier wave, resulting in a combination of both the carrier and information signal. We can think of the carrier signal as a vehicle for the base-band signal, which is being utilized in the channel. Because of modulation, it is possible to transmit multiple signals in a given band with avoiding interference, since carrier signal frequency can be chosen freely (but is higher than frequency of the original signal) [4]. Modulation is also useful in setting a high frequency for the transmitted signal, which makes it possible to use more compact antennas in signal transmission.

1.4.1 Pass-Band Signal to Complex Envelope

As it was mentioned in chapter 1.4, frequency of the carrier signal needs to be bigger than of the signal containing information that is to be transmitted. The information signal is usually a base-band signal, located at zero frequency [1]. Because of this, the modulation process will shift the signal for transmission to a higher frequency, more precisely at a pass-band frequency. This shifted signal is called band-pass signal and its spectrum can be seen at the top of Fig. 1.2. Because of the nature of Fourier's transformation, the band-pass signal has two symmetrical components at both positive and negative frequencies of the carrier [8]. The pass-band signal will be represented in time domain as $s(t)$ and is often called narrowband signal, because of its small bandwidth compared to its high frequency [6].

Analytical Signal

Signal is analytical when its spectral amplitude level is equal to zero at negative frequencies. It is created by combining the original signal with its Hilbert transform. Hilbert transform is a linear mathematical operation which creates a -90° and +90° phase shift on positive and negative frequencies respectively, of a real signal. Its application on the pass-band signal goes as:

$$s_a(t) = s(t) + js_h(t). \tag{1.2}$$

In frequency domain, analytical signal is mathematically explained with Hilbert transform as:

$$S_a(f) = S(f) + jS_H(f) = S(f) - j^2\mathrm{sign}(f)S(f) = S(f)(1 + \mathrm{sign}(f)) \tag{1.3}$$

$$\text{where} \quad 1 + \mathrm{sign}(f) = \begin{cases} 0, & \text{for } f < 0 \\ 2, & \text{for } f \geq 0. \end{cases} \tag{1.4}$$

From equations 1.3 and 1.4 its clear to see how the analytical signal loses its negative frequency component and gains double the energy on the positive one. [8]

Complex Envelope and I/Q data

Complex envelope can be gained by multiplying the analytical signal with $e^{-j2\pi f_c t}$:

$$s_e(t) = s_a(t)e^{-j2\pi f_c t}, \tag{1.5}$$

which corresponds in Fourier transform as a frequency shift by f_c:

$$S_e(f) = S_a(f + f_c). \tag{1.6}$$

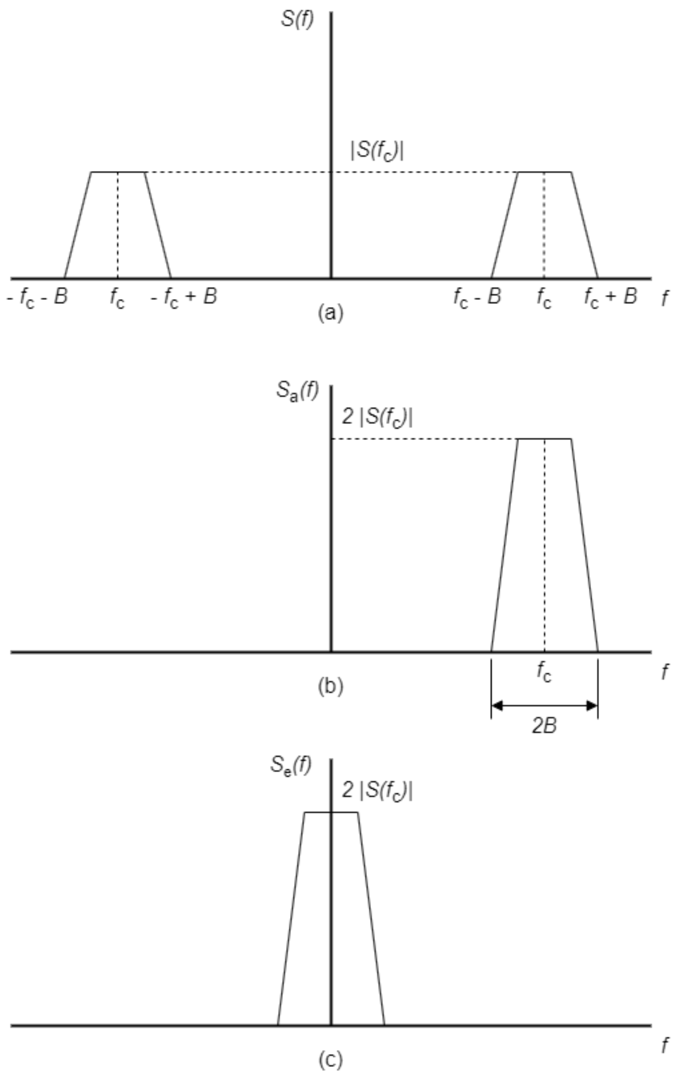


Fig. 1.2: (a) Band-pass signal, (b) Analytical signal, (c) Complex envelope (inspired by [6], [8])

Mathematically the chart in Fig. 1.2c represents now the spectrum $S_e(f)$ of the envelope from equation 1.6. This implies that the complex envelope $s_e(t)$ of the pass-band signal $s(t)$ is nothing more than a complex low-pass signal [6]. The pass-band signal is obtained as the real part of complex envelope multiplied by $e^{j2\pi f_c t}$. After inserting this notion into 1.5 it is possible to determine:

$$s(t) = Re\{s_e(t)e^{j2\pi f_c t}\}. \tag{1.7}$$

The complex envelope can be expressed in Cartesian form:

$$s_e(t) = s_I(t) + js_Q(t). \tag{1.8}$$

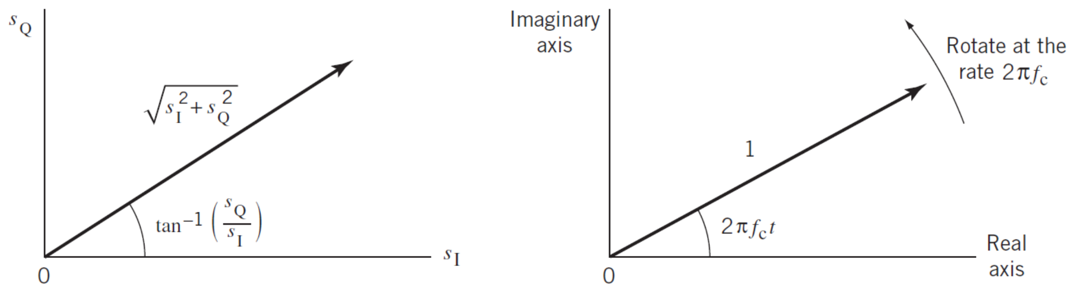

s_Q
$\sqrt{s_I^2+s_Q^2}$
$\tan^{-1}\left(\frac{s_Q}{s_I}\right)$
0
s_I
Imaginary axis
Rotate at the rate $2\pi f_c$
1
$2\pi f_c t$
0
Real axis

Fig. 1.3: Phasor rotation

When combining equations 1.7, 1.8 and using goniometric formulas, a standard equation for the original pass-band signal can be derived:

$$s_e(t) = s_I(t)cos(2\pi f_c t) + js_Q(t)sin(2\pi f_c t).[8] \tag{1.9}$$

The $s_I(t)$ is the real part of the complex envelope called in-phase component of the pass-band signal and $s_Q(t)$ is the imaginary part of the complex envelope referred to as the quadrature component of the pass-band signal. The complex envelope may be pictured as a time-varying phasor, which length and angle are given by both $s_I(t)$ and $s_Q(t)$ components as shown in Fig. 1.3. The plane where the phasor rotates is called the constellation diagram in which the in-phase component is represented along the horizontal axis and quadrature component is represented along the vertical axis. The points where the phasor aims are called symbols or I/Q data, often portrayed in the constellation diagrams. [6]

1.4.2 Digital Modulations

The use of digital modulations takes a bulk part of the practical work of this thesis. The previous chapter showed briefly the process of creating an amplitude modulation (AM), which is part of analog modulation's like frequency modulation (FM) and phase modulation (PM). When applying digital information from the original signal to the analog carrier, the modulated signal will fluctuate between certain states. This can be done by influencing the same parameters as in an analog carrier resulting in: amplitude shift keying (ASK), frequency shift keying (FSK) or phase shift keying (PSK) among many others. Keying refers to the changing of the modulated signal between the states. Those discrete states of the modulated signal are called symbols, explained in Chapter 1.4.1. In digital modulation those symbols are represented discretely. The coding of the symbols can vary depending on the modulation scheme. In M-ASKS, the modulated signals can achieve M different amplitudes, each representing a different symbol, while in M-PSK the information is not hidden in the absolute phase value, but rather in the change of the previous phase value by M possible phase shifts [8].

If 4-FSK is taken as an example, the modulated wave can achieve 4 discrete frequencies, and each of them is represented by 2 bits rather than 1 like in the case of 2-FSK. From this, the number of bits per symbol b, is given by

$$b = \log_2 M, \tag{1.10}$$

where M is the number of symbols. Other types of digital modulations include M-QAM modulations, which modulate amplitude and the phase as well. M-QAM modulations are often higher order modulations with multiple symbol like 16-QAM, 64-QAM and 256-QAM. A high number of symbols means they have more bits per symbol, therefore they exhibit a big data rate. In the practical part of this thesis, other variants of mentioned modulations are used, like CPFSK which, ensures continues phase transitions or GFSK which ensures smoother frequency transition. The used modulations also include PAM4 which is a used for analog signals with digital carriers, or WBFM which is a pure analog modulation with high noise resistance [4], [8].

1.5 Channel

A radio communication channel serves as the physical environment through which the signal is transmitted from the TX to the RX. In practical scenarios, various impairments impact the transmitted signal. The objective in signal processing is to describe as many of these impairments as possible and find appropriate approaches to mitigate their effects. To achieve this, impairments can be characterized by a range of parameters, such as Additive White Gaussian Noise (AWGN) and other channel models, fading, interference, etc. [4].

2 Machine Learning (ML)

The popularity of AI has exploded in recent years and has proven increasingly beneficial for various technology firms. This growth can be attributed to the development of more advanced computers, larger datasets, and new techniques in model training. Initially, AI excelled at solving problems that were considered difficult for humans, particularly those involving mathematical rules [9]. The true challenge for AI lay in solving tasks that are easy or even intuitive for humans, such as speech or image recognition [9]. Machine learning (ML) emerged as a solution to this challenge. This chapter aims to provide a basic understanding of ML, laying the groundwork for further exploration of the concept of Deep Learning (DL), which is integral to the practical theme of this thesis.

2.1 Deep Learning (DL)

Earlier, simpler ML methods faced a significant flaw as they heavily relied on the representation of input data. A major challenge for these methods came from real-world factors influencing the data. For instance, an image of a blue car might appear closer to black at night, the sun's reflection on the car wheels might render them undetectable, or the car might look different from various angles. DL-based methods offer a solution to these challenges [9].

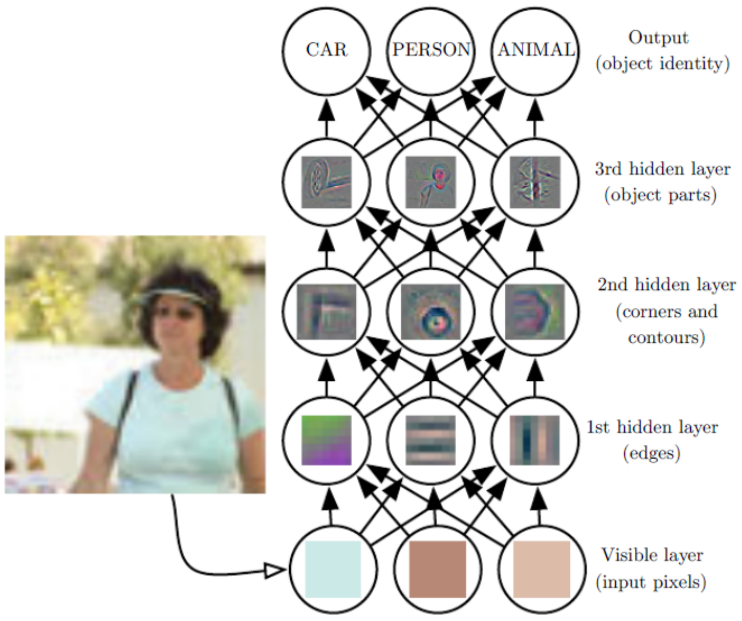


Fig. 2.1: Example of feedforward neural network [9]

A DL model offers a method for selecting important data and devaluing unnecessary data. It enables computers to construct complex concepts from simpler ones [9]. This involves the use of mathematical functions that map inputs to outputs, with these functions organized as neurons. Neurons are arranged into layers, and these layers collectively form a multilayer perceptron, also known as a feed-forward neural network. Fig. 2.1 illustrates a DL-based model tasked with object classification, composed of a multilayer perceptron.

As it is visible in Fig. 2.1, the multilayer perceptron divides the mapping task into a series of nested mappings, each represented by a different layer. In a multilayer perceptron, each neuron from one layer is connected to all neurons in the next layer (layers with these connections are termed Dense layers). The first layer is known as the visible layer, where variables observable by humans are input. Following this, the so called hidden layers are positioned. They are labeled '*hidden*' because, unlike the input layer, they do not directly interface with the raw input data, and unlike the output layer, they do not produce the final output [9]. These layers operate internally within the model, identifying relationships and patterns within the data. Their neurons compute output based on weighted input. These computations involve a parameter called weights, which are factors influencing the strengths of connections between neurons. This is a significant part of the optimization process, aiming to find the optimal weights of the DL model. In the case of a classification task (as in this thesis and also illustrated in Fig. 2.1), the output layer presents a result of the classification, usually in the form of attaching probabilities (calculated from weights) to its neurons. The neuron assigned with the highest probability indicates the classification result of the network.

2.2 Model Training Process

The goal of an ideal ML or DL model is to perform well on inputs on which the model is trained (training data) and also on new, unseen data (test data). The ability to perform well on unseen data is also referred to as model's generalization. With this requirement in mind, concepts such as training error can be introduced, which measures how well the model is performing on the training set. It is calculated by comparing the model's predictions with the actual target values in the training set. Test error, on the other hand, evaluates the model's performance on unseen data. The question arises: How can the model excel on unseen data? The answer is both sets cannot be collected arbitrarily from each other. It is necessary to adhere the so called i.i.d. (independent and identically distributed) assumptions. This assumption implies that the training set and the test set are both generated by a probability distribution in so called datasets. Furthermore, it is assumed that the examples in each set are independent of each other, and both sets have an identical distribution of data [9].

2.2.1 Overfitting and Underfitting

In practical scenarios, the training set is used to train the model and tune its parameters to minimize the training error. Consequently, the error of unseen data is expected to be greater or equal to the error of the set on which the model is trained. This understanding leads to two of the most significant challenges in ML: underfitting and overfitting. Underfitting occurs when the training error remains too high, indicating that the model is not learning effectively from the training data. Overfitting, on the other hand, is characterized by a substantial difference between the training and test errors, suggesting that while the model performs well on the training data, it fails to generalize effectively on new, unseen data [9].

How to control underfitting and overfitting of a model? The answer lies in altering model's capacity. Capacity can be roughly explained as the model's ability to capture and represent more complex data. Increasing capacity (or sometimes called complexity) in a DL model is therefore done by increasing the number of neurons and layers.

Figure 2.2 shows a typical relationship between capacity and error. As seen on the left side of this figure, training and generalization (test) error are both high. The model is not expressive enough to capture important information from the data. Eventually, after increasing the complexity of the model beyond the optimal capacity, the model starts to fit closely to random fluctuations of the training set, rather than training on true underlying patterns. This is where the overfitting zone is encountered.

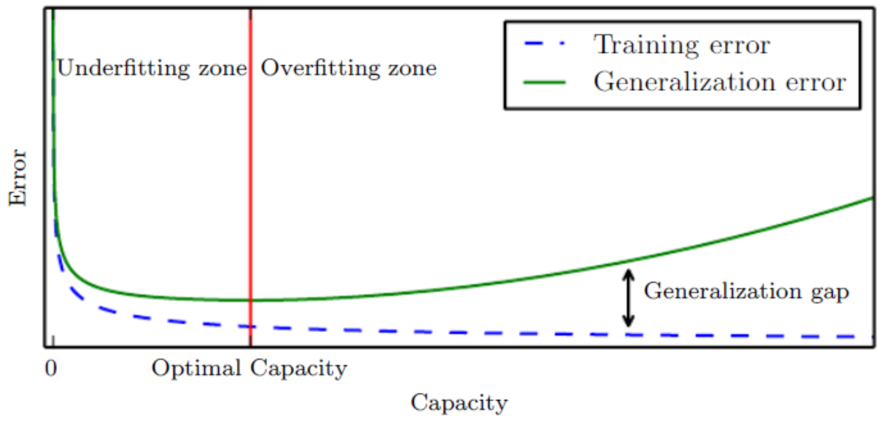


Fig. 2.2: Relationship between capacity and error [9]

Regularization Techniques

The method of implementing changes to the learning algorithm to reduce generalization error without influencing training error is called regularization [9]. One of simple methods to achieve this is by adding weight regularization, which adds a cost on the loss function for having a large weight. There are two ways to add regularization: L1 regularization where the added cost is proportional to the absolute value of the weight coefficient, and L2 regularization, where the added cost is proportional to the square of the weight coefficient value [10].

Another well-used technique for regularization in neural networks is called dropout. When dropout is applied to a specific layer, it involves randomly setting a number of output features to zero (effectively shutting down some neurons). This occurs only during training; during testing no neurons are dropped out, and instead, all outputs from the layer are scaled down by the dropout rate [10].

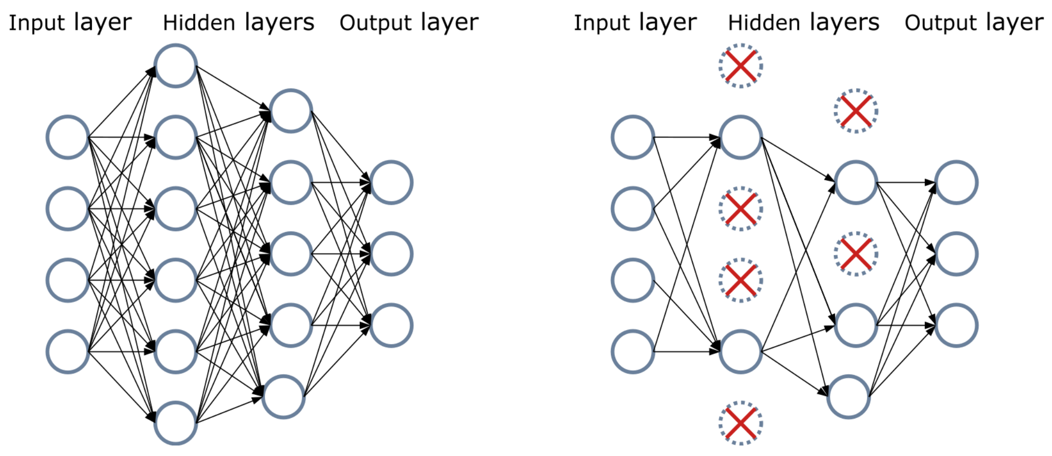


Fig. 2.3: Applying dropout on neural network [11]

2.2.2 Hyperparameters and Validation Set

Hyperparameters are settings that allow the control of the algorithm's behavior [9]. They are predetermined before the actual training and unlike other parameters such as weights, they are not adjusted by the model. Common examples of hyperparameters include the learning rate, the number of hidden layers and neurons, and the already mentioned regularization terms, among other specific to each model type. The primary objective is to set the hyperparameters in such a way, that minimizes the loss function. However, when hyperparameters are solely optimized on the training set, they would lead to a complex model with as much capacity as possible, resulting in overfitting [9].

For this reason, a separate validation set is employed. This set consists of examples not used in training and provides an unbiased evaluation of the model performance, helping in the tuning of the hyperparameters to ensure the model performs well on test data. The validation set is always constructed by splitting off a portion of the training set, usually at a ratio around 80% and 20% for training and validation data, respectively.

2.3 Convolutional Neural Networks (CNN)

By essence, CNNs are neural networks (NN) that have at least one convolutional layer. The difference between a convolutional layer and the previously mentioned Dense (fully connected) layer lies in several aspects. By structure, convolutional layers connect their neurons only to a small number of inputs, and this connectivity is determined by so called filters (or kernels). This allows the CNN to identify local patterns in input data, whereas a Dense NN focuses more on global patterns. Therefore, convolutional layers find their primarily application in tasks involving images or videos, where they can capture patterns such as edges, textures, or specific shapes [10].

In a CNN, the input data and the filter are convoluted with each other. Convolution is not described in detail here, but in the case of convoluting an input of a 2D (height, width) image, the process can be imagined as sliding a filter with its given size over the input data. In this way, the filter extracts patches from each position the filter goes through and calculates the dot product from each patch. The number of filters is a hyperparameter given by the user, let's mark it n. After all filters slide over each position in the input, all dot product results are built into a matrix called the feature map, with dimensions (height, width, n). The parameter n in the feature matrix is called the number of channels. This process is illustrated in Fig. 2.4. Please take into account that because of unmentioned hyperparamters stride and padding, the input feature map dimensions do not ressemble the expected output feature map dimensions [9], [10].

A very common procedure usually applied in CNN is putting the feature map through a pooling layer. This process aims to reduce the spatial dimension of the output feature map. By downsampling this output, the feature map coefficients are reduced, which intern reduces the parameters for the next convolutional or dense layer that follows after. A commonly used method is max-pooling, which takes the maximum value of the downsampled section. Another used parameter is called padding, which adds extra values on the edges of the feature map making sure the borders of the data are properly patched [10].

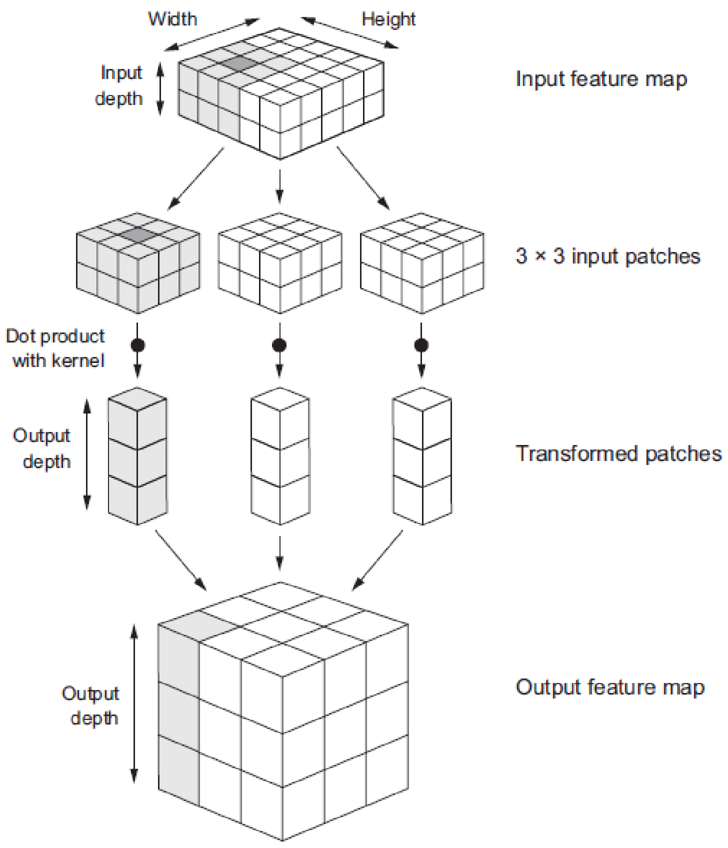


Fig. 2.4: Convolution in CNN [10]

3 Modulation Classification

Due to of the recent advancements in AI, the field of signal processing has also adapted this technique. Understanding parameters or labeling a received signal is crucial for processes like demodulation. Those classification tasks were often attributed to hand-crafting feature extractors [3]. Their task was seemingly very demanding due to various conditions affecting the channel, such as noise, fading or Doppler shift [1]. The classifiers thus have an important use, but it would be ideal if they did not require extensive knowledge of the transmission channel [1]. In the recent years, these traditional methods have rapidly given way to models with feature learning, such as ML and later predominantly DL. This chapter aims to demonstrate the basic use of DL in modulation classification.

The chosen programming language for the work is Python, which has established itself as the main platform for DL projects. With the emergence of libraries, such as Keras, TensorFlow, and Scikit-learn, the creation of code has become much easier. The used environment is Google Colab, a free cloud-based platform that offers virtual hardware often used for ML and DL. The platform facilitates data loading and saving directly from and to a Google Drive account. A proposed working process is shown on Fig. 3.1. The approaches and the subsequent challenges of this process are described in this chapter in detail.

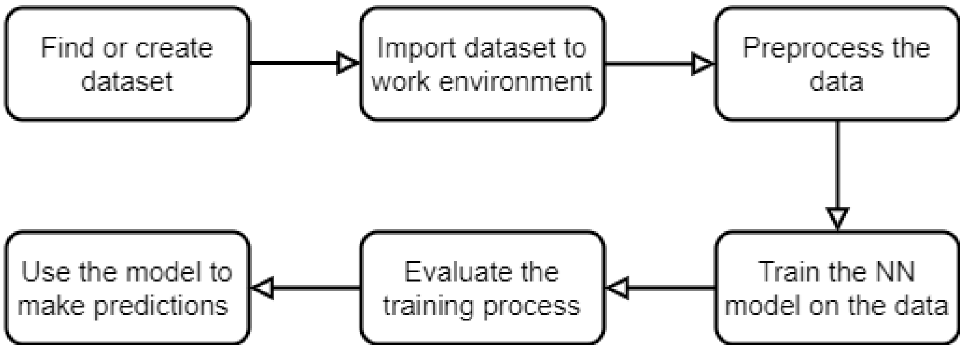

Find or create dataset
Import dataset to work environment
Preprocess the data
Train the NN model on the data
Evaluate the training process
Use the model to make predictions

Fig. 3.1: Working process

3.1 The Datasets

When training a DL-based model, ensuring the use of an appropriate dataset is crucial. The importance of a big amount of data cannot be overstated. As hinted in chapter 2.2, learning on a small amount of data may result into overfitting. Larger datasets offer a broader variety of data. Keeping this in mind, two suitable datasets have been selected for modulation classification in this thesis. The first one is an official dataset from the company Deepsig, often referenced in many works. The second one, as a part of this thesis, is a dataset generated in MATLAB.

3.1.1 RadioML Dataset

This dataset originates from the year 2016 when the company Deepsig began creating its own datasets. The dataset is synthetically generated using a software tool, further described in [12]. The version used in this work is 2016b. It contains 1200000 samples represented as 2x128 vectors of I/Q data of a modulated signal. These data are evenly split over ten different modulations: 8PSK, AM-DSB, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64, QPSK, and WBFM. Furthermore, the data are distorted using AWGN, fading models and multiple offsets. This added distortion overall sets the SNR range from -20 dB to 18 dB with an increment of 2 dB.

3.1.2 MATLAB Dataset

By utilizing MATLAB's Wireless Waveform Generator, a custom dataset was created. The structure of the samples is the same as in the RadioML Dataset, but now with only 160000 samples. Distortions of this dataset were generated only by adding AWGN and the SNR range is from -10 dB to 20 dB, again with an increment of 2 dB. The labelled modulations included are: 4PSK, 8PSK, 16QAM, 64QAM, and 256QAM. The modulation distribution looks plainer with only five labels, but, as will be proven, the choice of multiple QAM modulations will prove to be a quite difficult classification problem. The creation of the dataset was quite intuitive and the reason the same sample size was chosen as in RadioML was to make both datasets loading processes similar. In order to achieve the vector size of 128 symbols, care had to be taken because the application took input in bits. For this reason the equation 1.10 can be taken into account when calculating the needed input bits. In the case of 8-PSK, the number of bits per symbol is 3, hence multiplying 3 with 128 would give the total bits per symbol, and multiplying this results with the desired sample size for 8-PSK will result into the needed number of input bits.

3.2 Data Preprocessing

Preparing the data for model fitting is a crucial step in machine learning, as no level of model complexity or volume of data can compensate if the data is poorly interpreted by the algorithm. Firstly, the data had to be split into a feature set and a label set. The features are the I/Q data vectors, which had to stay aligned with their corresponding label (their modulation and their SNR values). Further, as mentioned in Chapter 2.2, this split data was distributed into the train, validation, and test set, with the ratio 70%, 15% and 15%, respectively. Since the data were divided randomly without any regards for the modulation and SNR values, this raises the question if some labels would not be underrepresented in the training. This is another reason for having a large dataset, the huge amount of data does not statistically allow for uneven distribution of data as seen in Fig. 3.2 and Fig. 3.3.

The next step is to apply one-hot encoding on the labels. One-hot encoding is a common method used to convert categorical variables into numerical form by assigning logical 1 to the correct label of the feature and logical 0 to other labels. It is often used in conjunction with the loss function applied in this model. Following label encoding, the next crucial step involves normalizing the input data across all sets. The purpose of this is to scale the data down to improve the performance of the model without changing the shape of the data. The used approach was Mean-Std normalization (standardization), which aims to transform the data so it centers around its mean value while maintaining a Gaussian distribution.

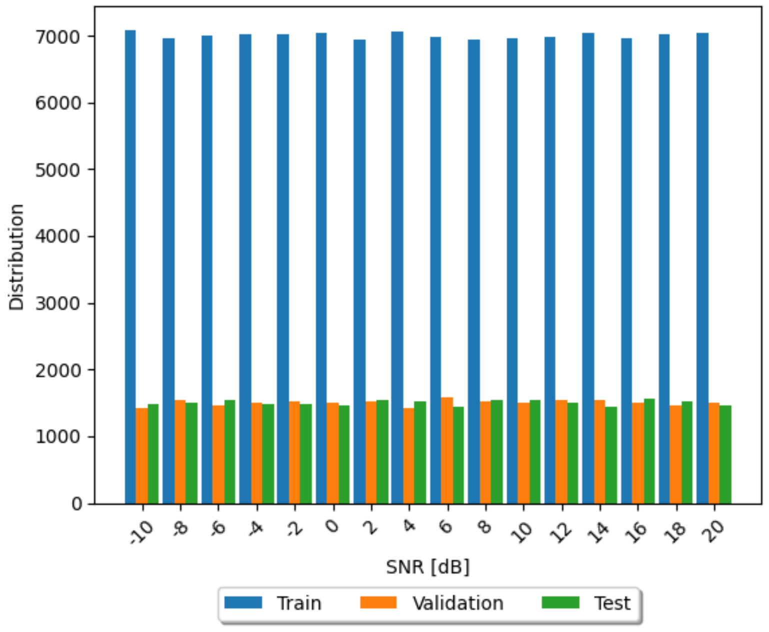


Fig. 3.2: SNR distribution across sets of MATLAB dataset

3.3 The Neural Network and Training

The used architecture is based on [1], where it was applied and tuned for modulation classification with a similar dataset. The input layer accepts each vector one by one,

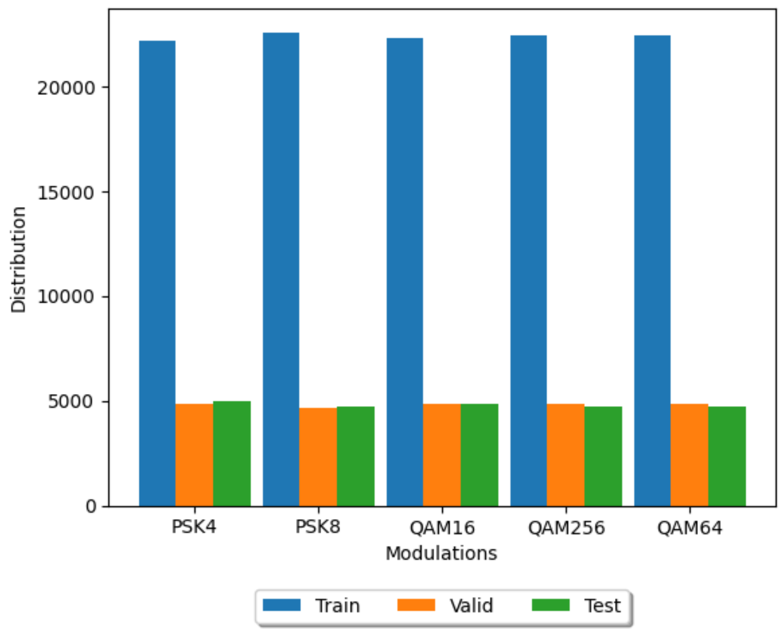


Fig. 3.3: Modulation distribution across sets of MATLAB dataset

and its output is framed with zeros at the zero-padding layer, allowing the kernel of the following convolutional layers to access the border regions of the data. There are exactly three convolutional 1D layers, meaning the kernel moves only across one dimension, outputting only a 2D feature map (length and channel). The activation function used in those layers was the rectified linear activation function (ReLu). After each of them follows max-pooling, with an additional Dropout at the end. Next, a flatten layer follows with the task to convert the data dimensions to a 1D vector, compatible for the dense layers following after, first activated by scaled exponential linear unit fucntion (SeLu) and the second activated by the Softmax function, which transforms the neuron weighs into 10 probabilities (in the case of the Deepsig dataset) or 5 probabilities (in the case of the MATLAB dataset). The described architecture is summarized in Table 3.1 and on Fig. 3.4.

Other parameters and their settings that are not included in Table 3.1 are the learning rate, which was set to 0.0007 at the start of training but could adaptively tweak up to 0.00007 in case of no improvement in validation loss after 3 training epochs. The loss function Categorical Crossentropy was used to calculate the validation loss. Early stopping was also applied which would stop the training process if validation loss did not improve after 5 epochs. The batch size of the model was set to 64, decreasing after each training process by half. After each passage the weights were updated by the Adaptive Moment Estimation optimizer (Adam optimizer). However, these settings were not optimal for the Matlab dataset, not even

Table 3.1: Architecture and hyperparameter setting of the CNN [1].

Layer	Hyperparameters	Output Shape
Input	-	128x2
Zero padding	Padding 4	136x2
Conv 1D	Filters = 50, Kernel Size = 8, ReLu	129x50
Max Pooling 1D	Pool size = 2	64x50
Conv 1D	Filters = 50, Kernel Size = 8, ReLu	57x50
Max Pooling 1D	Pool size = 2	28x50
Conv 1D	Filters = 50, Kernel Size = 4, ReLu	25x50
Dropout	0.6	25x50
Max Pooling 1D	Pool size = 2	12x50
Flatten	-	600
Dense	70 Units, SeLu	70
Dense	10 or 5 Units, Softmax	10 or 5

after tuning of some of the mentioned hyperparameters. The model had an accuracy of 20% which effectively meant that the model was guessing since exactly 5 classes were present. The highest accuracy that was achieved was 25% which afterwards did not increase and the training process was stopped by early stopping. Therefore it was decided to cut out the samples with negative SNR values. After this the model finally fit to the Matlab dataset, but with a learning rate of 0.00007.

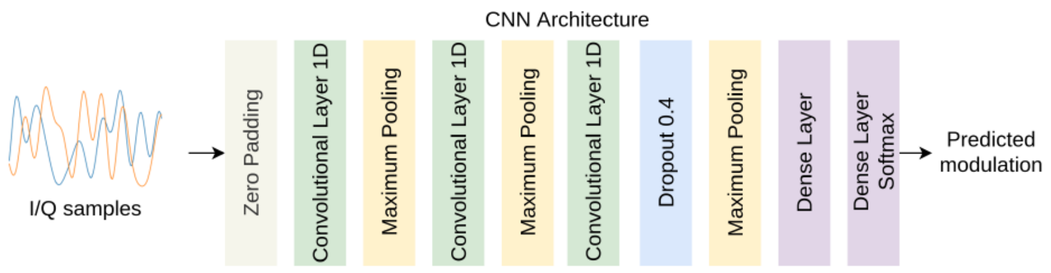


Fig. 3.4: Visualised CNN architecture [1]

3.4 Results

After the training process the model was used to predict classes from the test features. Those prediction were then compared with their corresponding labels of the test data. The results are presented in the form of cunfusion matrices. Confusion matrices are a commonly used tool in supervised machine learning, their rows represent the actual classes, while the columns represent the predicted classes in the same sequence. They display accuracy, which is the proportion of examples that were correctly classified by the model, so the opposite of error [9]. Therefore high probabilities on the main diagonal indicate better results. The confusion matrices for both datasets are displayed on Fig. 3.5 and Fig. 3.6. Please note that when predicting, the model took input features across the whole test set without taking account for the SNR values.

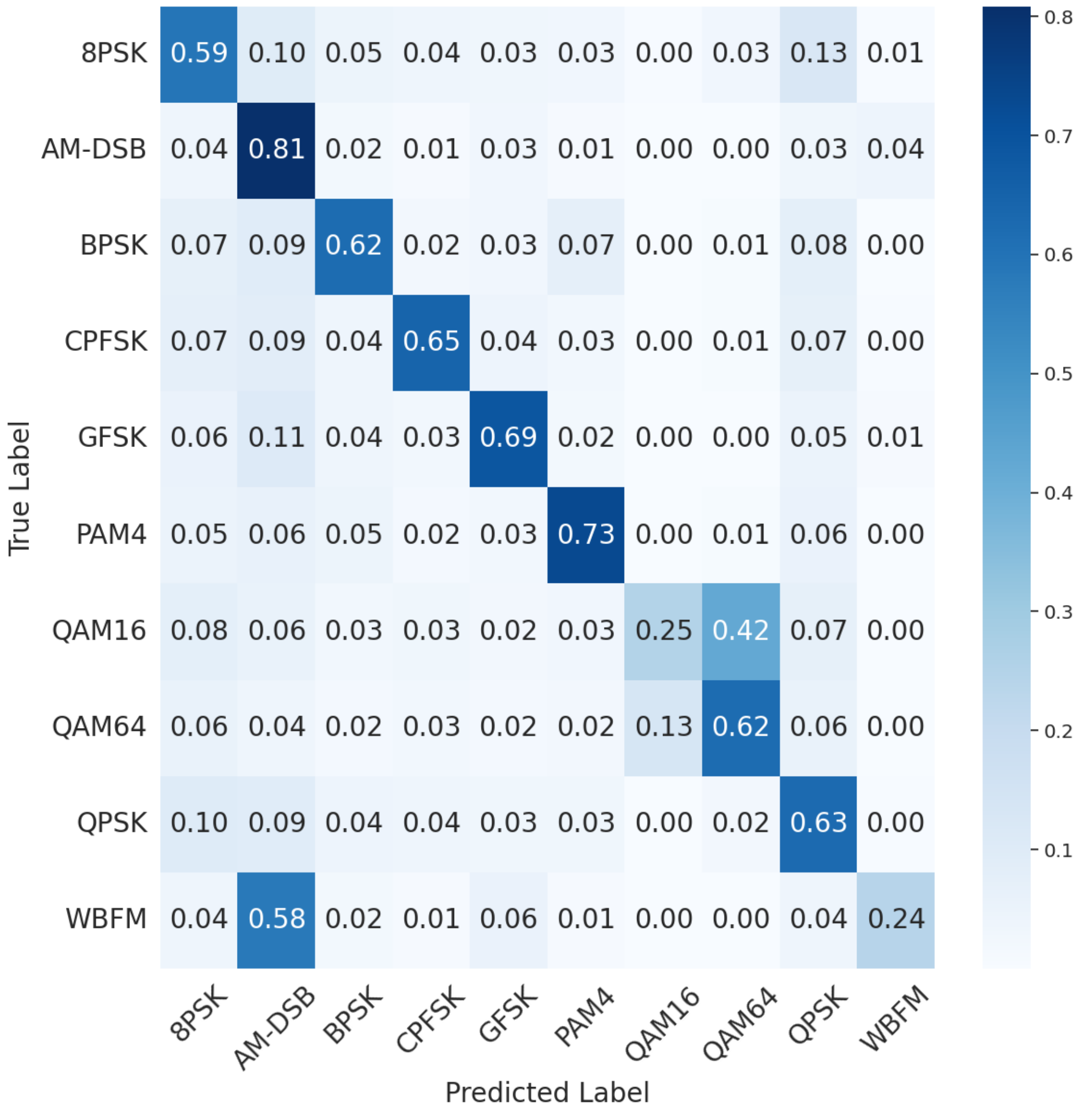


Fig. 3.5: Confusion Matrix over all SNR values of DeepSig dataset

As can be seen on both figures the model had a particular difficulty in distinguishing schemes of M-QAM modulations. Those modulations are high-order modulations, they can carry more bits per symbol which results in a higher data rate as mentioned in Chapter 1.4.2. However, when taking a look at the constellation diagrams of 16-QAM, 64-QAM and 256-QAM, there would be a similarity in the I/Q data values since higher-order M-QAM modulations are more densely occupied, making them more prone to impairments. Maybe because of max-pooling, which truncates the output feature map, the model was not able to find patterns in the amount of symbols which could have been useful when classifying those modulations. An other type of NN may have been more suitable for this issue.

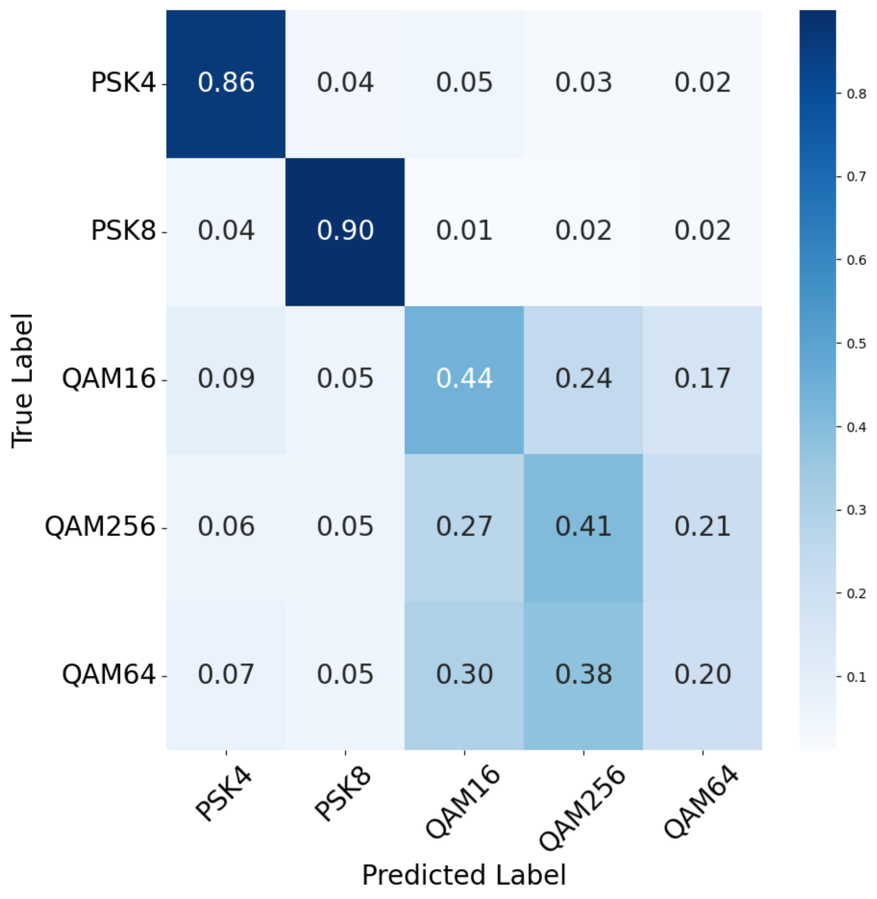
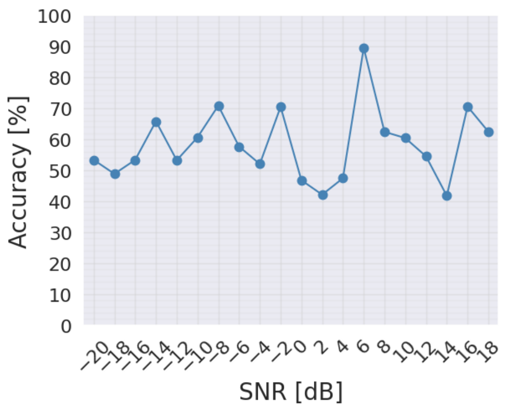


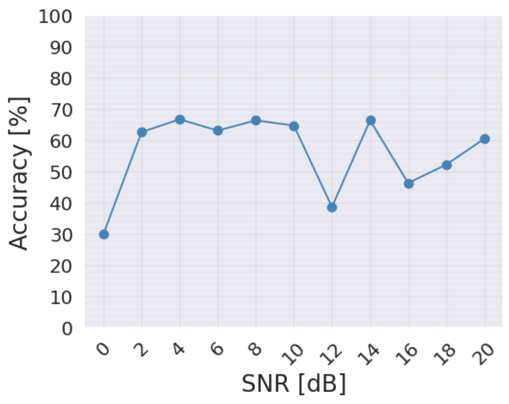
Fig. 3.6: Confusion Matrix over all SNR values of Matlab dataset

There is also a known error in the DeepSig dataset that confuses the classification of WBFM for AM-DSB. It arises because of the voice samples being quite short which causes some samples to record only a pause in the speech, that leading into the transmission of pure carrier signals, which is the same for the only two analog modulations in this set (AM-DSB and WBFM) [1]. That is pretty much a pity, because WBFM has very good SNR resistance and having classification clarity over it could have been an indication of how much the results got influenced by noise.

Other results are better but leave much to be desired. By first glance an accuracy of 60% at other labels may not be that bad, but when taking a look on Fig. 3.7 it is interesting to note that no big progressive improvements over better SNR samples are observed. This is a strong indicator that the model did not fit data as it was meant to, it was not optimized to the global minimum but rather converged to a local minimum. This can be also seen in Fig. 3.7b where a sudden accuracy drop is observed on a higher SNR value. The overall testing accuracy for DeepSig dataset was 58.25% and for the truncated Matlab dataset 56.19%.



(a) For DeepSig dataset



(b) For Matlab dataset

Fig. 3.7: Accuracy across SNR

Conclusion

The assignment for this thesis was to provide possibilities in classification of RF signals via ML or DL. The research was to be provided on different signal types, sent under diverse circumstances (like noise influence, fading, etc.). Free online datasets were to be utilized, with an option to create an own dataset. An approach to train those datasets on an ML or DL algorithm was to be proposed.

At the beginning of the thesis, an introduction to signal processing is described, with an emphasis on communication systems, particularly on modulation. The process of pass-band modulation is more briefly described, which provides an introduction to I/Q data and digital modulations, terms widely used in the practical part of this thesis. In the next theoretical part, the core concepts of ML and DL are explored. Key terminology is explained, occasionally delving into details. The understanding of those details in DL proved to be instrumental in practical parts of this thesis. However, information provided in this chapter serves merely as an introduction. Aspiring DL designers will require much more understanding to excel in this field. The last part introduces the reader to the practical work, undertaken for this thesis. It presents a straightforward workflow, which is further elaborated in the subsequent sections. The chapter tries to chronologically explain the proceeded approaches for the used steps that were taken and the challenges that emerged. Subsequently, it presents and comments on the outcomes of predictions of both models.

The used NN was a CNN which originally stems from [3] and was tuned in [1] (particularly the kernel size parameter). The results for now provide a good understanding and a groundwork for further research in this thesis. However, it is fair to say now that the given CNN did not converge to an optimal solution, which is strange, considering the CNN performed better in other works that included the DeepSig dataset. On the other hand, both models were able to make classifications not solely based on guessing, which is interesting to observe, that a different solution was found under seemingly similar settings. The plan for the next phase of the practical work is to find a setting for models to improve on the underlying patterns of the I/Q data on both datasets and to make a model fitting the whole Matlab dataset. Those goals lie in extensive tuning of the hyperparametes, maybe with the use of sophisticated methods like grid search. This processes will require further research into DL theory. Another aim is to expand from the current modulation classification task to different classification tasks, like RF signal type recognition transmitted under various conditions. In pursuing these goals, other types of NNs may be explored (like recurrent neural networks), tested and compared with each other.

Bibliography

[1] Kristyna Pijackova and Tomas Gotthans. Radio modulation classification using deep learning architectures. Bachelor's thesis, Brno University of Technology, The Faculty of Electrical Engineering and Communication, 2021. URL: `https://www.vut.cz/en/students/final-thesis/detail/133594`.

[2] Joseph T. Downey, Benjamin C. Hilburn, Tim O'Shea, and Nathan E. West. Machine learning remakes radio. *IEEE Spectrum*, 57:35–39, 2020. URL: `https://api.semanticscholar.org/CorpusID:216587284`, `doi:10.1109/MSPEC.2020.9078454`.

[3] Timothy James O'Shea, Tamoghna Roy, and T. Charles Clancy. Over-the-air deep learning based radio signal classification. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):168–179, 2018. `doi:10.1109/JSTSP.2018.2797022`.

[4] Václav Žalud. *Moderní radioelektronika.* BEN - technická literatura, Praha, 1. vyd. edition, 2000.

[5] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. `doi:10.1002/j.1538-7305.1948.tb01338.x`.

[6] Simon S. Haykin. *Digital Communication Systems.* Wiley, 1 edition, 2013.

[7] Basab Purkayastha and Kandarpa Sarma. *A Digital Phase Locked Loop based Signal and Symbol Recovery System for Wireless Channel.* 01 2015. `doi:10.1007/978-81-322-2041-1`.

[8] Roman Maršálek. *Teorie rádiové komunikace.* Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních . . . , 2012.

[9] Aaron Courville Ian Goodfellow, Yoshua Bengio. *Deep Learning [pre-pub version].* MIT Press, 2016.

[10] François Chollet and J. J. Allaire. *Deep learning with R / François Chollet ; with J.J. Allaire.* Manning Publications, Shelter Island, New York, 2018.

[11] Wikimedia Commons. File:neural network dropout.svg — wikimedia commons, the free media repository, 2022. [Online; accessed 23-December-2023]. URL: `https://commons.wikimedia.org/w/index.php?title=File:Neural_Network_Dropout.svg&oldid=720153962`.

[12] Timothy O'Shea and Nathan West. Radio machine learning dataset generation with gnu radio. *Proceedings of the GNU Radio Conference*, 1(1), 2016. Available: Dataset available at: `https://www.deepsig.ai/datasets`. URL: `https://pubs.gnuradio.org/index.php/grcon/article/view/11`.

Symbols and abbreviations

A/D	Analog to Digital Converter
AI	Artificial Intelligence
AM	Amplitude Modulation
AM-DSB	Amplitude Modulation with Double Sideband
ASK	Amplitude-Shift Keying
AWGN	Additive White Gaussian Noise
CNN	Convolutional Neural Network
CPFSK	Continuous-Phase Frequency-Shift Keying
D/A	Digital to Analog Converter
DL	Deep Learning
FM	Frequency Modulation
FSK	Frequency-Shift Keying
GFSK	Gaussian Frequency-Shift Keying
I/Q	In-Phase/Quadrature
ML	Machine Learning
PAM	Pulse Amplitude Modulation
PM	Phase Modulation
PSK	Phase-Shift Keying
QAM	Quadrature Amplitude Modulation
RF	Radio Frequency
RX	Receiver
NN	Neural Network
SNR	Signal-to-Noise Ratio
TX	Transmitter

WBFM	Wideband Frequency Modulation