Juha-Matti Inkilä

OFDM CHANNEL ESTIMATION WITH MACHINE LEARNING

Faculty of Information Technology and Communication Sciences
Bachelor's thesis
Examiner: Taneli Riihonen
May 2022

ABSTRACT

Juha-Matti Inkilä : OFDM Channel Estimation With Machine Learning
Bachelor's thesis
Tampere University
Information technology
May 2022

Orthogonal frequency-division multiplexing (OFDM) is a method of encoding data on multiple subcarrier frequencies and is commonly used in today's wireless communication systems. One of its many advantages is that it allows for frequency-domain equalization of the signal at the receiver removing the need for complex time-domain equalizers. The channel equalization process consists of estimating the adverse effects that the channel has on the signal, such as fading and distortion, and minimizing their contribution. Channel estimation in OFDM systems is usually implemented as a pilot-aided estimation, where known data is sent through the channel periodically. Zero-forcing (ZF) and minimum mean square error (MMSE) estimators are traditional low-complexity algorithms used in pilot-aided schemes. As of late, machine learning approaches have been a growing topic of interest in OFDM performance and link quality research.

This thesis examines the possibility of using a deep neural network in determining and estimating the true transmitted data after propagating through a noisy channel. These estimates are used to reduce bit-error rate (BER) and to achieve better signal quality. The performance of the machine learning estimator is compared against traditional ZF and MMSE estimators in terms of BER as a function of signal-to-noise ratio (SNR).

The machine learning (ML) estimator is taught using data produced by a MATLAB simulation of an OFDM transmitter and receiver. The estimator receives 16-QAM modulated symbol's in-phase and quadrature coefficients, and a least-squares spline-interpolated estimate of the channel's frequency response. It then performs the channel equalization implicitly, outputting a Gray-encoded prediction of the true transmitted symbol. The results of this study show that the ML estimator can at least match the performance of conventional zero-forcing and MMSE estimators, producing a BER/SNR -graph similar to them.

Keywords: OFDM, Machine Learning, Channel estimation, Channel equalization

TIIVISTELMÄ

Juha-Matti Inkilä : OFDM-kanavan estimointi koneoppimismenetelmän avulla
Kandidaatin tutkielma
Tampereen yliopisto
Tietotekniikka
Toukokuu 2022

Ortogonaalinen taajuusjakoinen multipleksointi (OFDM) on tiedonsiirtomenetelmä, joka perustuu tiedon koodaamiseen usealle alikantotaajuudelle, ja se on hyvin yleisesti käytetty tekniikka nykyisissä langattomissa tiedonsiirtojärjestelmissä. Yksi sen monista eduista on, että se mahdollistaa signaalin korjauksen taajuustasolla vastaanottimessa, vähentäen tarvetta monimutkaisille aikatason ekvivalisaattoreille. Kanavan ekvivalisointiprosessilla tarkoitetaan kanavan signaaliin aiheuttamien vääristymien ja häipymien arviointia, sekä niiden vaikutusten minimointia. Kanavan estimointi OFDM-järjestelmissä toteutetaan yleensä pilottiavusteisena arviointina, jossa tunnettua dataa lähetetään sovituin välein. Nollapakotus (ZF) ja minimikeskimääräisen neliövirheen (MMSE) arviointialgoritmit ovat perinteisiä matalan kynnyksen menetelmiä pilottiavusteisissa järjestelmissä. Viime aikoina koneoppimisen hyödyntäminen OFDM-suorituskyvyn ja viestikanavien laadun parantamisessa on ollut kasvava kiinnostuksen aihe kommunikaatiojärjestelmien tutkimuksen alalla.

Tämä työ tutkii syväoppivan neuroverkon hyödyntämistä kanavan estimointiin ja todellisen lähetetyn datan päättelyyn. Päättelyn avulla pyritään vähentämään kanavassa tapahtuvaa bittivirhesuhdetta (BER) ja saavuttamaan parempi signaalin laatu. Syväoppivan järjestelmän suorituskykyä verrataan tavanomaisempiin nollapakotus ja MMSE-estimaattoreihin BER / signaalikohinasuhde (SNR) -käyrien avulla.

Koneoppiva estimaattori opetettiin MATLAB OFDM -simulaatiosta saadulla datalla. Koneoppiva järjestelmä saa tiedon vastaanotetun 16-QAM moduloidun symbolin in-phase ja quadrature -komponenteista sekä interpoloidun raakatiedon kanavan taajuusvasteen arviosta kyseiselle symbolille. Tämän jälkeen järjestelmä tekee kanavan ekvivalisoinnin implisiittisesti ja tuottaa Gray-enkoodatun arvion todellisesta lähetetystä symbolista. Tutkimuksen tuloksien perusteella koneoppiva järjestelmä kykenee vähintään vastaamaan ZF- ja MMSE-estimaattorien suorituskykyä, tuottaen vastaavanlaisen BER/SNR-käyrän.

Avainsanat: OFDM, koneoppiminen, kanavan estimointi, kanavan ekvivalisointi

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AWGN	Additive white Gaussian noise
BER	Bit-error rate
CFR	Channel frequency response
ICI	Inter-carrier interference
ISI	Inter-signal interference
LS	Least squares
LOS	Line of sight
MMSE	Minimum mean square error
ML	Machine learning
OFDM	Orthogonal frequency-division multiplexing
QAM	Quadrature amplitude modulation
ReLU	Rectified linear unit
RX	Receiver
SNR	Signal-to-noise ratio
TX	Transmitter
WLAN	Wireless local area network
ZF	Zero-forcing

1. INTRODUCTION

Many of today's wireless communications systems including 802.11a/g/n WLAN, LTE networks and terrestrial digital TV DVB-T/H systems use the orthogonal frequency-division multiplexing (OFDM) scheme for data encoding and transmission. A multi-user access variation of OFDM called OFDMA is also used extensively in modern 4G cellular networks. Like any real communications signal propagating through a channel, an OFDM signal also experiences distortion and fading effects caused by irregularities in the environment. Such irregularities might be for example obstacles blocking the signals direct path, therefore breaking the line-of-sight (LOS) between the signal transmitter (TX) and the receiver (RX). The result is multipath propagation, a phenomenon where the signal finds many alternative pathways between the TX and the RX. Different components of the signal arrive at separate times and sum up at the receiver to create errors in the data transmission. To counteract these effects, channel equalization is needed. Its purpose is to nullify, or at least minimize, the adverse effects of the channel and ideally make the channel frequency response flat across all frequencies.

In recent years, plenty of research has been done on the utilization of machine learning methods for OFDM transmissions [1] [2]. These approaches show great promise for the future in terms of improving communications reliability and link quality for OFDM systems. For example, the joint project between Tampere University and Nokia Bell Labs - DeepRX, a convolutional deep learning receiver, has shown to outperform traditional approaches for receivers and their ability to perform channel estimation in certain settings [3].

The goal of this thesis is to study the possibility of using a machine learning (ML) based channel estimator at the receiver. The ML estimator's performance will be compared against traditional estimation algorithms, namely zero-forcing (ZF) and minimum mean square error (MMSE), in terms of bit-error rate (BER) as a function of signal-to-noise ratio (SNR). The training data for the ML estimator will be generated using a MATLAB based OFDM simulator. The performance tests will also be performed on the MATLAB simulation.

The test results show that the ML estimator's performance matches the conventional estimators' performance in cases with 8 and 16 pilot subcarriers. All three of the estimators stay very close to one another, and no clear winner can be picked in any of the scenarios. The ML estimator seems to be most restricted by its very limited input features, as it only bases its prediction on the least squares channel estimate. More raw data could allow it to make more novel predictions.

Chapter 2 discusses the relevant background topics, such as basics of OFDM systems and channel estimation techniques. Chapter 3 details the architecture of the deep learning model used for the ML channel estimator. The MATLAB simulation and the details of the OFDM scheme used are detailed in Chapter 4 along with the experiment results. Chapter 5 concludes with the observations and discusses potential future work and improvements for the model.

2. BACKGROUND

Certain key aspects, such as the basics of wireless OFDM data transmission and channel estimation/equalization, are discussed in this chapter.

2.1 Orthogonal frequency-division multiplexing (OFDM)

OFDM has many benefits over single-carrier schemes, with the main attraction being its robustness against frequency-selective fading and narrowband interference [4, p. 11]. Other notable advantages are high spectral efficiency, ease of dealing with delay spread and other multipath effects and the possibility of using single-frequency networks in broadcasting applications [4, p. 14].

The main idea of OFDM is to use multiple equally spaced subcarrier frequencies to carry the data. Each subcarrier is assigned a modulated data symbol. The frequency difference between each carrier frequency is called the subcarrier spacing.

The orthogonality between subcarriers means that two OFDM subcarriers do not cause any interference to each other after demodulation. This is the case even if the spectrum of neighboring subcarriers clearly overlap, as can be seen from figure 2.1. The lack of interference between the OFDM subcarriers is not simply because of the subcarrier spectrum separation. Rather, it is due to the specific frequency-domain structure of each subcarrier in addition to the specific choice of a subcarrier spacing Δf equal to the per-subcarrier symbol rate $\frac{1}{T_u}$. [5, p. 35]

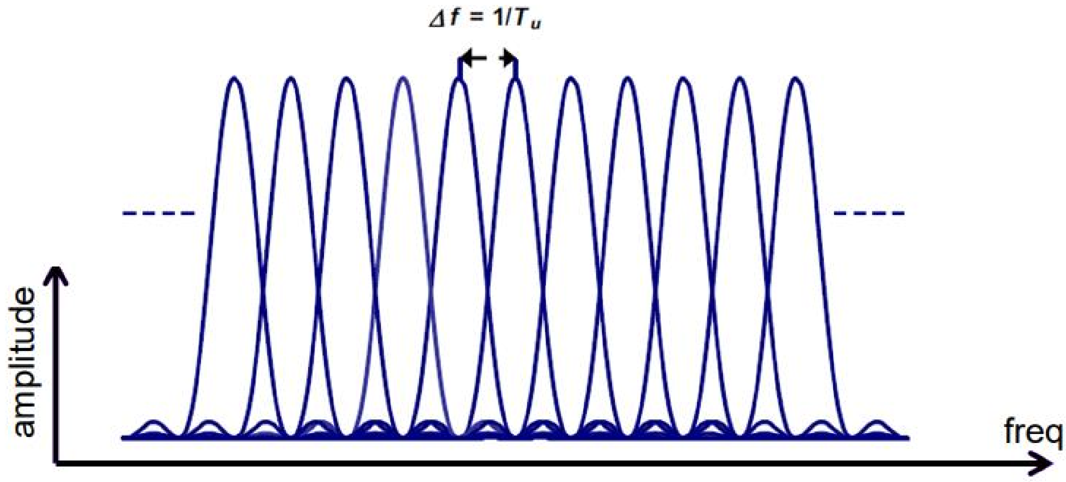
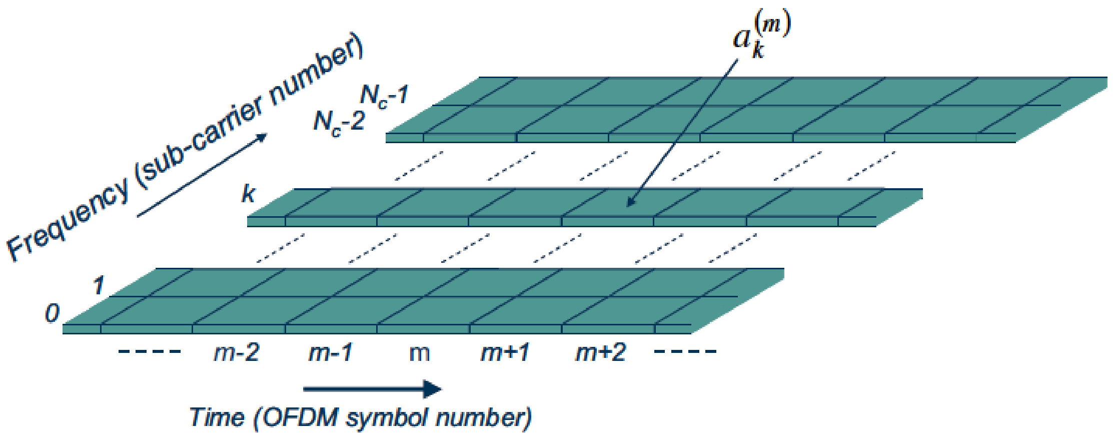


Figure 2.1. An example of OFDM subcarrier spacing in frequency domain. The subcarrier peaks correspond to nulls of adjacent subcarriers, resulting in zero inter-carrier-interference (ICI). Adapted from [5, p. 30].

Any distortion of the frequency-domain structure of the OFDM subcarriers, for example due to a frequency-selective channel, may lead to a loss of the orthogonality and to inter-carrier-interference (ICI). To prevent this and to make an OFDM signal robust against frequency selectivity, cyclic-prefix insertion is typically used. [5, p. 32] Cyclic prefix insertion principle is discussed in more detail in 2.2 Multipath propagation and frequency-selective channels. One OFDM symbol can consist of N amount of subcarrier frequencies. Once an OFDM symbol has been constructed from the subcarriers, they can be sent consecutively in a block-like fashion to form a data stream, as figure 2.2 demonstrates.



$a_k^{(m)}$
Frequency (sub-carrier number)
N_c-1
N_c-2
k
1
0
---- m-2 m-1 m m+1 m+2 ----
Time (OFDM symbol number)

Figure 2.2 A time-frequency grid resembling multiple OFDM-symbols in the time-domain axis, and their subcarriers in the frequency axis. One column of subcarriers forms one OFDM-symbol. Adapted from [5, p. 32].

For example, a typical WLAN-transmission might use anywhere from 52 to 64 subcarriers in one OFDM symbol. Modern 4G-LTE mobile communications systems may use up to 2048 different subcarriers to ensure high performance, but also for the use of multi-user schemes, where different subcarriers are assigned to different users.

2.2 Multipath propagation and frequency-selective channels

When a signal propagates through a medium in the real world, it faces all sorts of obstacles on its path. These cause the signal to reflect off and bend around said obstacles, scattering around their environment randomly. Some of these scattered signals find alternative paths to the receiver, and multiple versions of the sent signal arrive at the receiver with varying delays. These signals then sum up at the receiver, on the premise of the superposition principle, causing interference and degradation of signal quality. This phenomenon is called multipath propagation interference. [6, p. 7] Multipath propagation interference is best combated using the cyclic prefix insertion in OFDM systems. A portion, defined by the cyclic prefix size, of subcarrier content at the end of one OFDM symbol is copied over at the beginning of the symbol sequence in the transmitter. This content is then discarded at the receiver. This way, if the cyclic prefix in the time-domain persists for longer than the channel delay spread, the effects of multipath interference are negated, and zero inter-signal-interference (ISI) is achieved. Figure 2.3 shows this idea visually. τ_{max} is the maximum multipath delay spread of the channel, which the cyclic prefix should be able to cover.

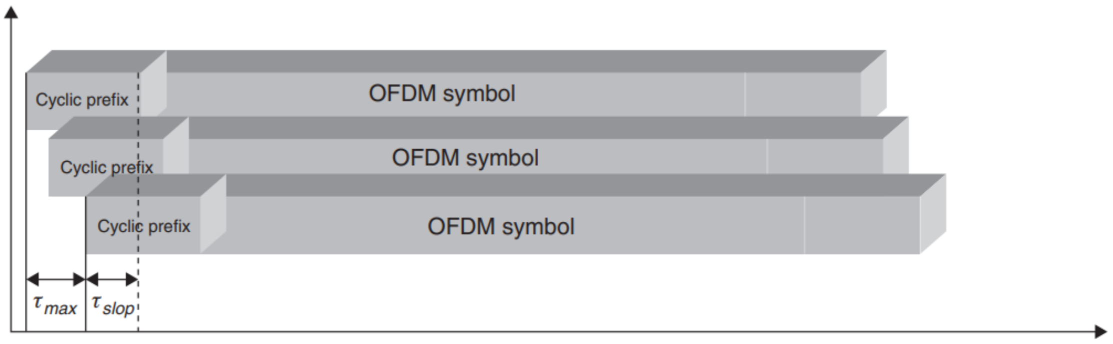


Figure 2.3 Multiple OFDM-symbols overlapping due to multipath propagation. Adapted from [7, p. 70].

When the cyclic prefix lasts longer than the channel delay spread, it ensures that the received signal holds all the channel multipath effects, and a channel estimate is at least theoretically able to restore the signal to its original shape [7, p. 70]. In essence, the cyclic prefix tries to prevent two consecutive OFDM symbols from overlapping in the time-domain and thus causing distortion of the signal [8, p. 39].

2.3 Channel estimation

Usually, cyclic prefix insertion alone does not completely remove the adverse effects a channel might have on the signal. If a channel is frequency-selective, it will distort an OFDM signal by attenuating the amplitude and offsetting the phase of each subcarrier frequency varyingly. These effects can cause errors in the data transmission. The way to mitigate this is to estimate the channel response and equalize the received carrier contents accordingly. Thus, the goal of channel estimation and equalization is to restore the original shape of the signal to some extent [5, p. 49]. A popular scheme to estimate the channel state information in OFDM systems is to use pilot symbols. In pilot-aided schemes known data is sent either periodically or on fixed frequencies, and the estimation is based on the difference between the sent and recovered data. [8, p. 107] Several traditional estimation algorithms exist, while this study concerns itself with two of the most basic and prominent ones – zero-forcing and minimum mean square error estimators.

2.3.1 Zero-forcing estimator

For a frequency-selective channel a ZF estimator will give the frequency-domain equalizer coefficients as follows

$$C(f) = \frac{1}{H(f)} , \tag{2.1}$$

where $C(f)$ is the equalizer coefficient and $H(f)$ is the channel frequency response for frequency f. In essence, a zero-forcing equalizer multiplies the channel response with its reciprocal, yielding a flat frequency response and linear phase. A ZF equalizer is simple to implement, and it is particularly useful in cases where ISI is significant compared to noise.

2.3.2 Minimum mean square error estimator

MMSE estimators work the same as ZF in high SNR scenarios. However, they generally produce better results in low SNR scenarios, since they use additional channel information in their estimation – mainly the channel noise variance. The frequency-domain equalizer coefficients for the MMSE are given by

$$C(f) = \frac{(H(f))^*}{|(H(f))|^2+\sigma^2} , \tag{2.2}$$

where $(H(f))^*$ is the conjugate of the channel frequency response for frequency f and σ^2 is the channel noise variance [8, p. 109]. Sometimes the noise variance is not known directly and will also need to be estimated through other means.

3. METHODOLOGY

The experiments were performed on a MATLAB implementation of an OFDM transmitter-channel-receiver model. The ZF and MMSE estimators are coded natively in the MATLAB language, while the machine learning estimator is written in Python using TensorFlow libraries. The machine learning estimator is trained using data generated by the simulator. When performing channel equalization, the MATLAB code calls a Python script which interfaces with the ML estimator. The implementations of both the MATLAB code and ML estimator are discussed in more detail in this chapter.

3.1 MATLAB OFDM simulator

A flowchart of the simulator is presented in figure 3.1, which includes all the main processes involved in sending and receiving one OFDM-symbol.

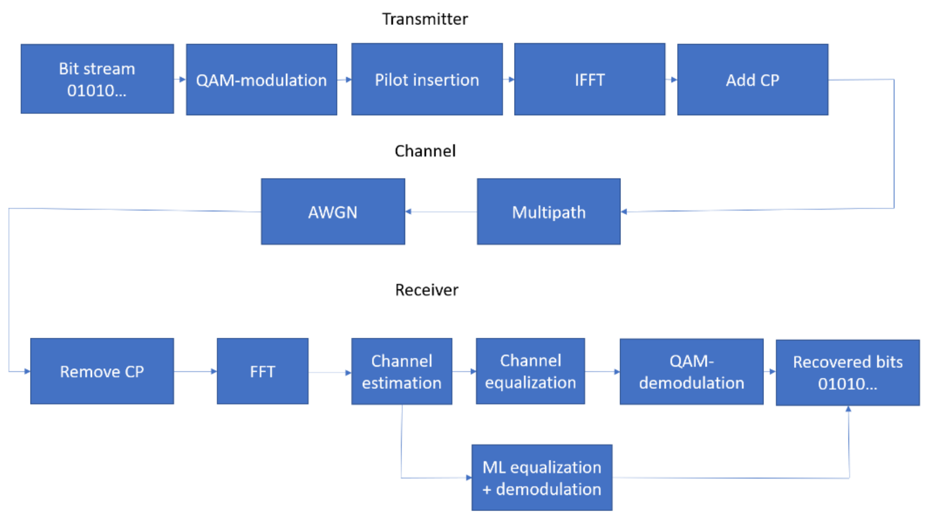


Figure 3.1 A flowchart presentation of the MATLAB simulator sending and receiving one OFDM-symbol.

When transmitting an OFDM symbol, the simulator first generates *k* * *N_Data* randomized bits, where *k* is the binary logarithm of the M-QAM modulation order and *N_Data* is the amount of data subcarriers used. The bits are then modulated into quadrature amplitude modulated (QAM) symbols using the Gray-coding scheme. Next, these data symbols are assigned to data subcarriers, and pilot symbols to pilot subcarriers in a comb-type manner as illustrated in figure 3.2. An inverse fast Fourier transform is then performed to move from frequency-domain to time-domain. Finally, a cyclic prefix of length $n * T_{sym}$ is inserted at the beginning of the OFDM time-domain symbol.

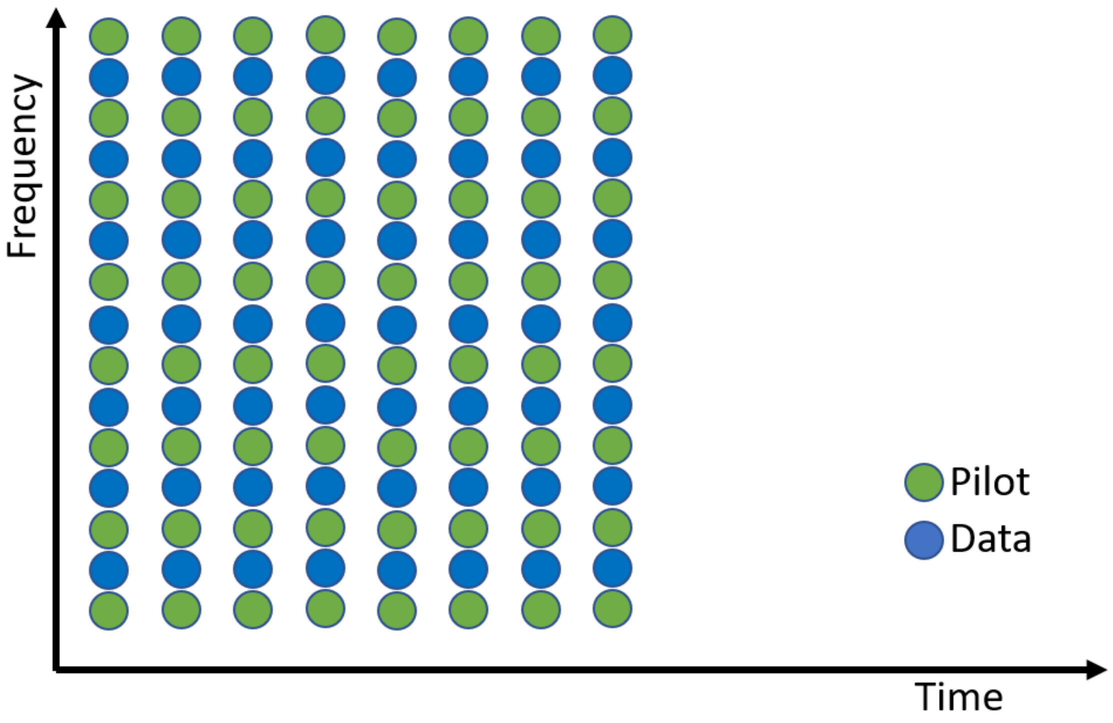


Figure 3.2 An illustration of a comb-type pilot insertion.

The signal experiences a channel, which is simulated by convolving the time-domain signal with a multipath channel filter with four taps and adding additive white Gaussian noise (AWGN). The channel coefficients are randomized for each OFDM symbol.

The receiver discards the cyclic prefix and performs fast Fourier transform to move back into the frequency domain. A least squares channel frequency response estimation is performed at the pilot frequencies. This estimation is then spline-interpolated across the whole channel and the received symbols are equalized using the ZF (2.1) and MMSE (2.2) techniques. The simulator also calls a Python script, which interfaces with the neural network. The neural network performs the channel estimation, equalization and demodulation of the symbols implicitly. The operation of the neural network is presented in more detail in sections 3.2 and 3.3. The symbols are demodulated, converted back into bits and the amount of bit errors is calculated for each equalization technique: no equalization, ZF, MMSE, ML and full channel knowledge.

The simulation can be run for a range of SNRs while sending multiple OFDM symbols each time to plot a BER/SNR chart. The chart can be used to compare the performance of each equalization technique.

3.2 Feedforward neural network

A feedforward neural network is the simplest kind of a deep learning network. Its structure typically consists of an input layer, hidden layer(s) and an output layer [9]. The basic structure is visualized in figure 3.3.

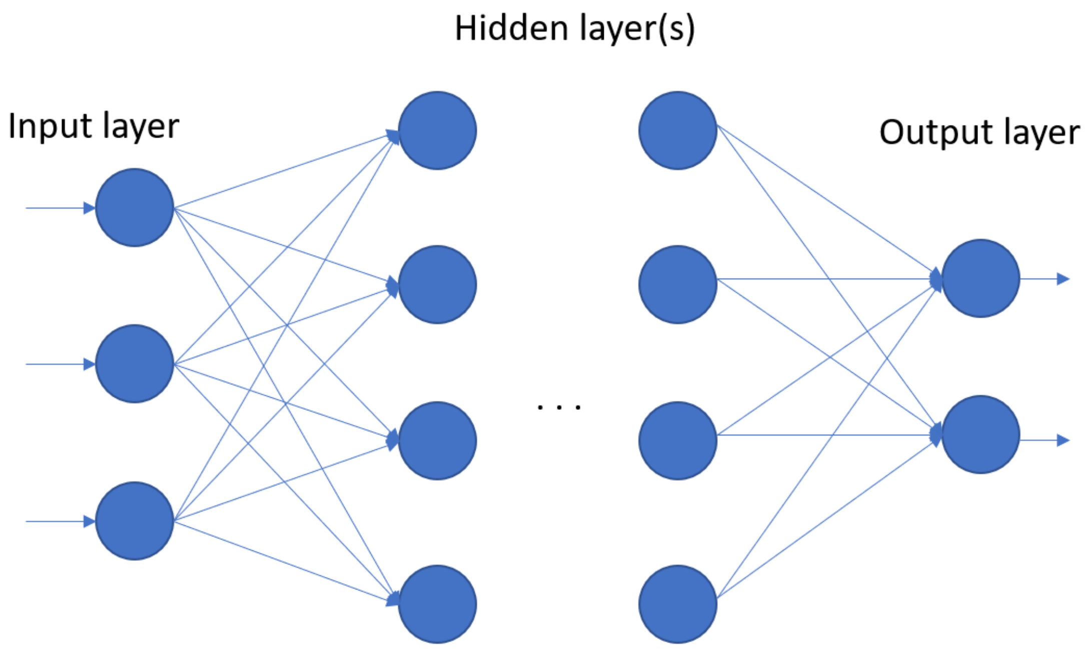


Figure 3.3 An illustration of a feedforward neural network.

Each layer consists of multiple perceptrons, which are the building blocks of the network. A perceptron is a logical unit, whose operation resembles that of a mammalian neuron [10]. Each perceptron consists of input synapses, a summation function and an activation function. The structure of a perceptron is illustrated in figure 3.4.

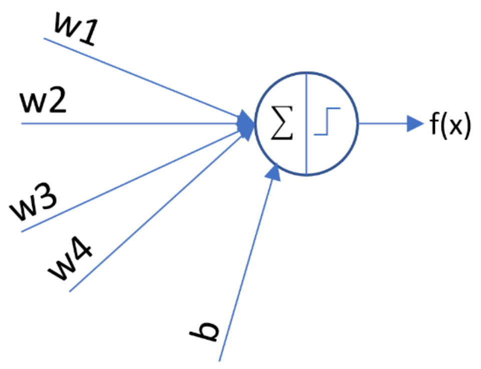


Figure 3.4 An illustration of a perceptron with four input weights.

In a feedforward network, a perceptron receives its input from preceding perceptrons. These inputs are then multiplied with a corresponding weight *w_k* and summed together using the summation function. An additional adjustable bias term *b* is added to the sum. The sum is then fed into an activation function, which determines the output of the perceptron. The activation function brings non-linearity to the perceptron's output, as it could otherwise only be used to produce linearly separable functions. One common non-linear activation function is the rectified linear unit (ReLU) characterized by

$$f(x) = \max(0, x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases}. \tag{3.1}$$

When a feedforward network is used for multiclass classification, its output layer will have perceptrons equal to the number of classes. The network will classify its input based on which output perceptron will have the highest activation value. Usually in these cases the final output activation function used is softmax, characterized by

$$f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}, \tag{3.2}$$

where z_i is the output of perceptron *i* and *j* is the number of perceptrons in the layer. Softmax essentially normalizes the outputs of the perceptrons in the interval *[0, 1] and* ensures the components will add up to 1. Thus, the layer's output can be treated as a probability distribution, where the highest output corresponds to the most likely class.

In supervised learning the network learns by calculating the error of its outputs compared to the ground-truth labels and adjusts its weights and biases through gradient descent. The calculation of the error is determined by the network's loss function. The network will try to minimize its loss function, and therefore approximate a function, which best fits the training data.

One commonly used loss function in classifying problems is the cross-entropy loss function defined as

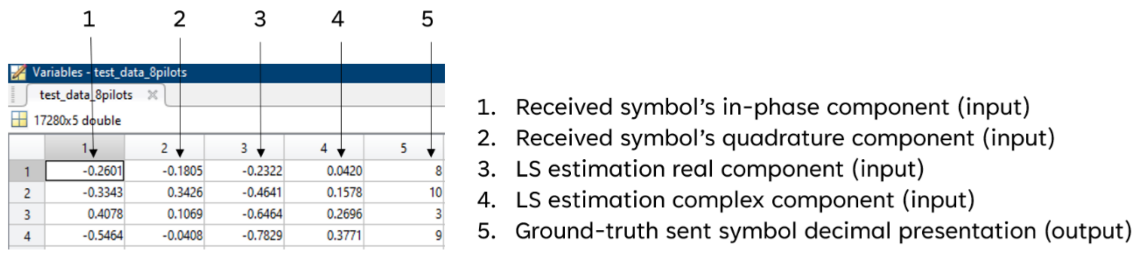
$$L = -\sum_i^C y_i * log\widehat{y_i}, \tag{3.3}$$

where y_i is the true ground-truth label for the class *i*, $\widehat{y_i}$ is the predicted label for class *i,* and *C* is the total amount of classes. Categorical cross-entropy requires the output labels to be one-hot encoded, where each cell in a vector of length *C* corresponds to one class label. Sparse categorical cross-entropy is a variation of categorical cross-entropy, where the labels need not to be one-hot encoded. Sparse categorical cross-entropy fits use cases where each of the classes are mutually exclusive.

3.3 OFDM ML equalizer

The proposed ML equalizer is a feedforward neural network consisting of one hidden layer that is 64 wide and an output layer of size 16. The hidden layer has ReLU as its activation function and softmax in the output layer. The model uses the Adam optimizer for learning, with the learning rate set to 0.001, and sparse categorical cross-entropy as it's loss function.

The training data was generated using the OFDM simulator. The network's inputs are the received QAM quadrature and in-phase components, and the interpolated least squares channel estimate. The network is trained to classify the sent symbol's gray-coded decimal presentation based on the input data. Thus, the network performs the channel estimation, equalization and symbol demodulation implicitly, yielding the sent data directly – apart from the decimal to bits conversion. The network's inputs and outputs are visualized in figure 3.5.



	1	2	3	4	5
1	-0.2601	-0.1805	-0.2322	0.0420	8
2	-0.3343	0.3426	-0.4641	0.1578	10
3	0.4078	0.1069	-0.6464	0.2696	3
4	-0.5464	-0.0408	-0.7829	0.3771	9

1. Received symbol's in-phase component (input)
2. Received symbol's quadrature component (input)
3. LS estimation real component (input)
4. LS estimation complex component (input)
5. Ground-truth sent symbol decimal presentation (output)

Figure 3.5 An example of the test data used in training for the evaluation of the ML equalizer.

The training data was split into an 8:2 ratio of training data and validation data, and the model was trained for 20 epochs with a batch size of 64. An epoch means that all the training data is used once for training. Multiple epochs mean multiple runs with the training data.

Models for both 8- and 16 pilots in an OFDM symbol were created. Figure 3.6 shows the training accuracies, and the loss function values for both models. The training and test data accuracies for both models are also shown in table 3.1.

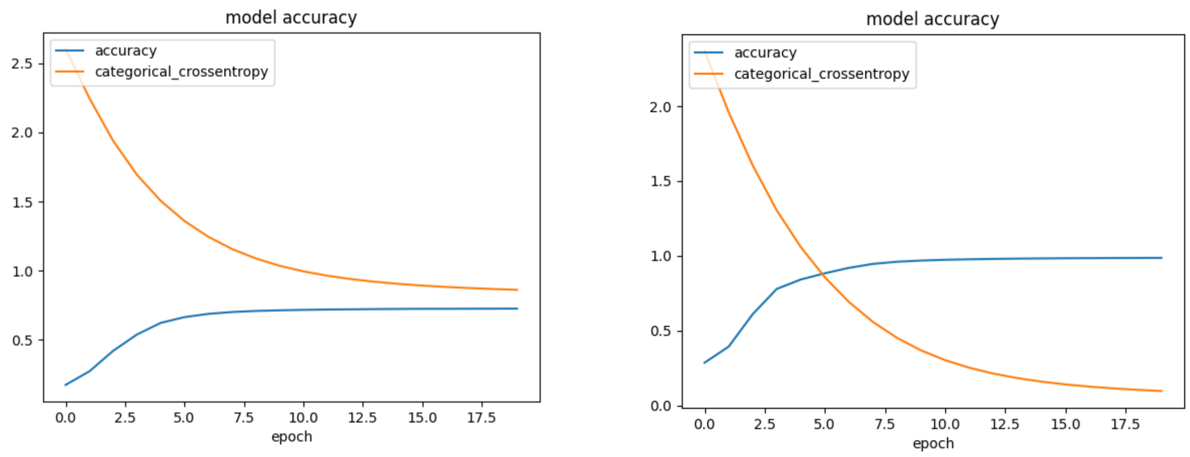


Figure 3.6 Training model accuracy for 8-pilot model (left) and for 16-pilot model (right). The accuracy and loss depicted is for the training data.

Table 3.1 Training and test data accuracies for 8-pilot and 16-pilot ML equalizer models. Accuracy means how many sent symbols the model classifies correctly.

Model	**Training accuracy (SNR 25-30)**	**Test accuracy (SNR 1-30)**
8-pilots	~73.10%	~58.35%
16-pilots	~98.85 %	~77.80 %

The 8-pilot model was trained with 2682 samples of OFDM symbols in the SNR range of 25 to 30. The 16-pilot model was trained with 15630 sample OFDM symbols also in the SNR range of 25 to 30. Both models were tested with a test data of 270 OFDM symbols in SNR range 1 to 30.

4. EXPERIMENTS

The experiments were performed on the OFDM simulator by incrementing the SNR in the range *[1, 30]* and sending 10^5 bits, or approximately 446 OFDM symbols, per each increment. After each successfully sent OFDM symbol, the amount of bit errors was calculated for each equalization technique, and the total amount was tracked. A BER/SNR chart was then generated from the data. From the chart, it is possible to compare the performance of each estimator/equalizer in terms of how many bits they get right for each SNR. Two experiments were conducted, where the amount of pilot subcarriers was first set to 8 and then 16. The M-QAM modulation order was set to 16 for all experiments and the total amount of subcarriers to 64. The simulator also assumed that the cyclic prefix was able to cover the channel delay spread completely. The OFDM scheme variables are presented in table 4.1.

Table 4.1 Mutual variables used in the simulator OFDM scheme for both experiments.

FFT size	Total Subcarriers	QAM order	CP length	Channel taps
64	64	16	8	4

The results show that with 8 pilot subcarriers the ML equalizer can match the performance of the traditional methods. The performance of each method is very close to one another, and they all settle around the 10^{-1} BER range in high SNR. The overall results for 8 pilot subcarriers are depicted in figure 4.1.

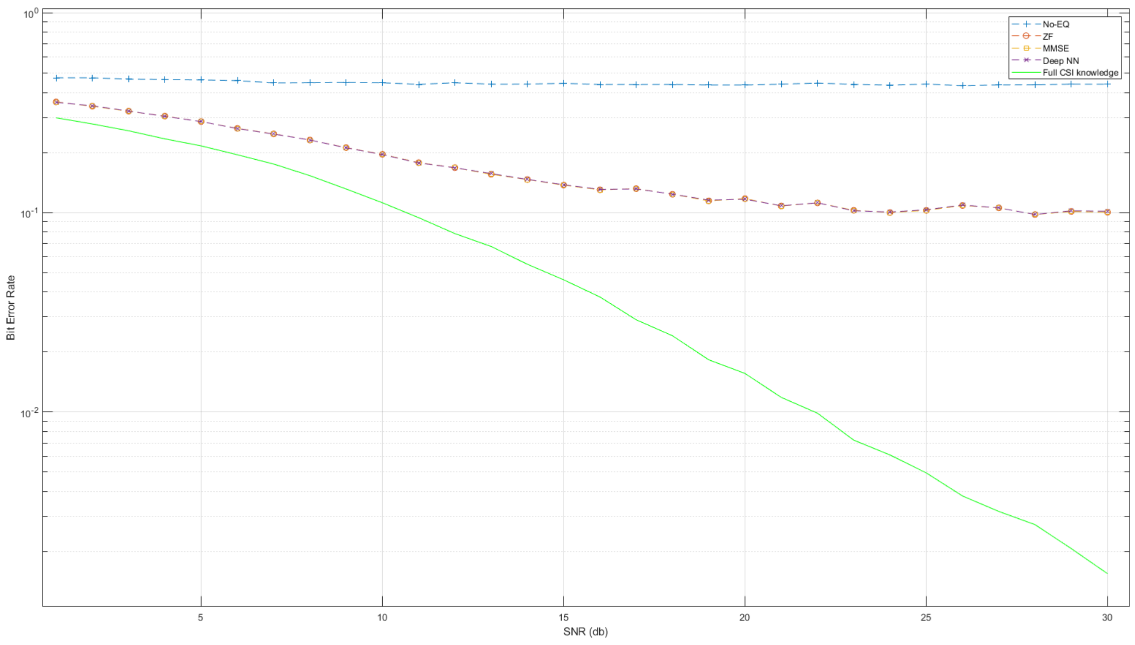


Figure 4.1 BER/SNR chart for 8 pilot subcarriers for each OFDM symbol in SNR range *[1,30].*

The ML equalizer performance oscillates over and under the ZF and MMSE performance, sometimes performing better and sometimes worse. The varying in performance can be explained as random fluctuation, and no real competitive difference can be derived from this dataset.

In the case of 16 pilot subcarriers, the ML equalizer performs quite similarly as the traditional methods and achieves a similar BER for the same SNR. The overall results for 16 subpilot carriers are depicted in figure 4.2.

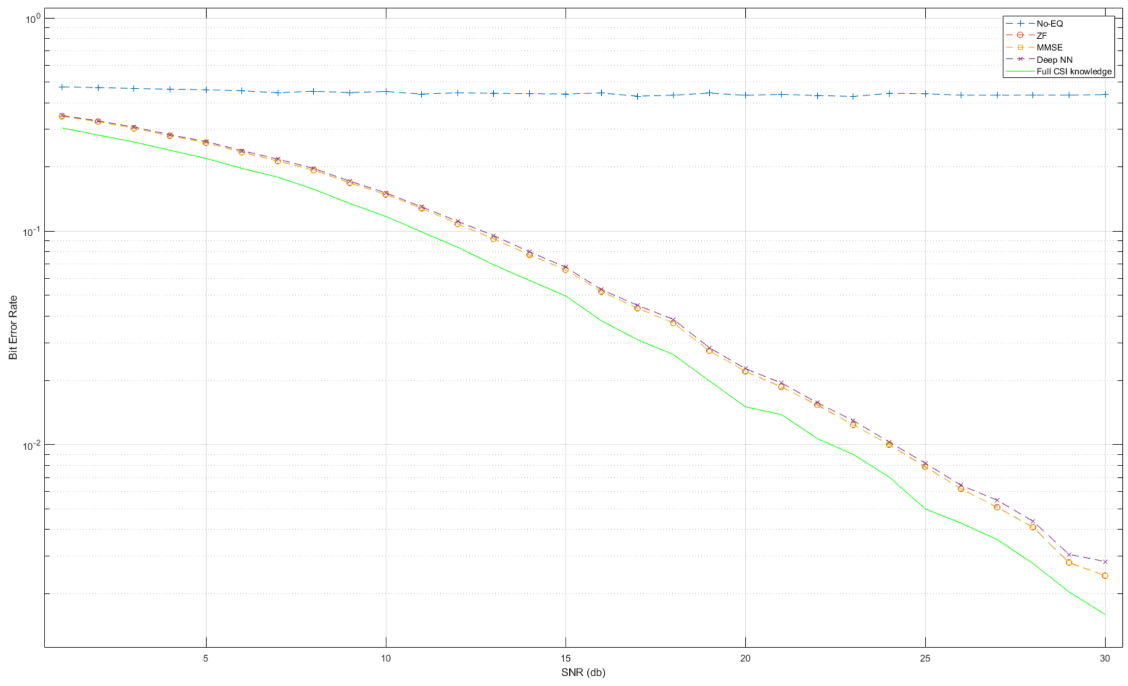


Figure 4.2 BER/SNR chart for 16 pilot subcarriers for each OFDM symbol in SNR range *[1,30].*

The 16-pilot model however, unlike the 8-pilot model, never actually produces better results than the traditional methods ZF and MMSE. This can be explained with the problem of overfitting, where the machine learning model learns the training set very accurately but does not generalize well into real world test data [11]. From table 3.1 it can be noted that the 16-pilot model gets an almost perfect accuracy (98.85%) on the training set, while the test accuracy remains under 80% (77.80%), indicating overfitting. In addition, since the only input data the model uses for its estimation is the LS interpolated channel estimate, and the LS gives an almost perfect channel estimate with 16 pilots in high SNR, could it contribute to the overfitting problem. The models were trained with high SNR samples, and in the 16-pilot model this seems to lead to overconfidence of the model. An example of the estimation of channel equalization coefficients given by ZF and MMSE estimators for 16 pilots can be seen in figure 4.3.

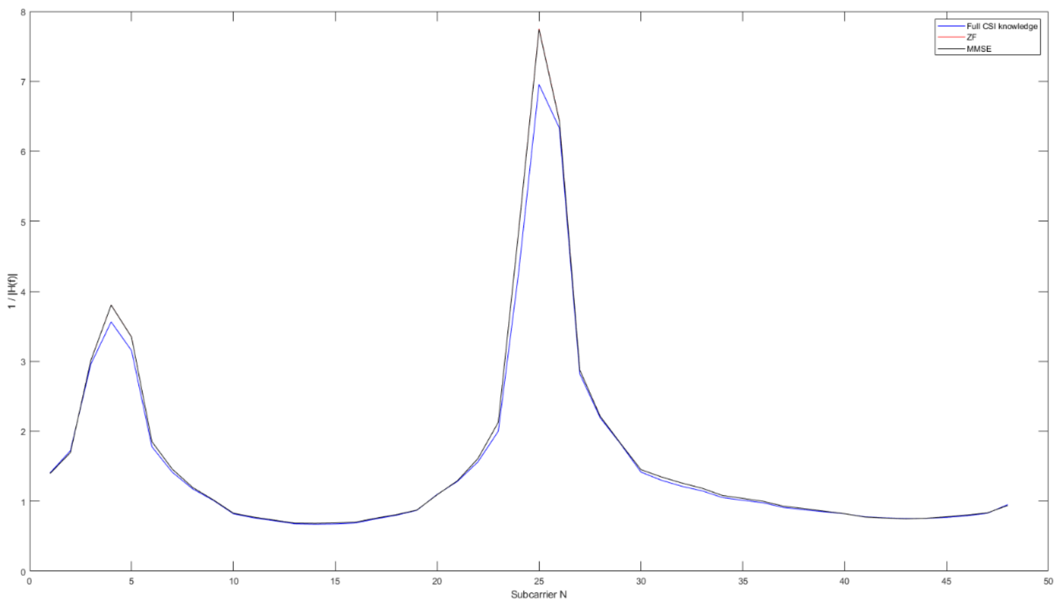


Figure 4.3 An estimation of channel frequency response equalization coefficients for one OFDM symbol with 16 pilot subcarriers and 30 SNR. Only the data subcarrier indices are considered.

In high SNR and with 16 pilots the traditional estimators get very close to the ideal equalization coefficients, which might lead to the ML model fine-tuning its layer weights too explicitly during training. The use of validation data during training does not seem to prevent this from happening.

5. CONCLUSIONS

The goal of this thesis was to experiment and examine the possibility of using machine learning in OFDM channel estimation and equalization. The model suitable for the scenario and type of data set up by the OFDM simulator ended up being a simple feedforward classifier neural network. The main restriction the simulator sets for choosing and building a model for the machine learning approach, is the fact that the channel is completely randomized for each OFDM symbol. Thus, it is not possible to gain any channel information from consecutive OFDM symbols. A scenario the OFDM simulator most closely depicts is a WLAN data transmission with some multipath elements, with only one OFDM symbol sent at a time. All the code and data used for this thesis are available at GitHub[1].

The ML equalizer clearly performs better than no equalization at all and is somewhat competitive with the ZF and MMSE methods, producing a similar BER/SNR curve. Its greatest challenge comes from the fact that it relies too heavily on the LS channel estimation, since it is the only additional feature the estimator uses as its input. Perhaps a more interesting approach would have been, if the estimator was given only the received OFDM symbol, sent pilots and the pilot indices. In this case, the classifier would have had to come up with a novel way of performing the channel estimation on its own, while not relying on any form of preprocessing of the data. In its current form, it is reasonable to postulate that it in fact learned the ZF estimator and behaves in an analogous way. Any gains it might have over the traditional methods come from the fact that it yields the sent symbols directly, effectively performing some form of decision demodulation in the process. The ML equalizer can also be estimated to have a low time complexity due to the small size of the network.

Overall, the idea of using a classifier for solving this type of a problem seems to work well. The model could be improved upon by using 1-D convolutional layers for higher feature extraction resolution, or remodeling it into a recurrent neural network, to make use of the temporal sequences an OFDM signal transmission might have.

[1]https://github.com/JooHis/OFDM-simulator-with-ML-equalizer

6. REFERENCES

[1] Shahzad, H., Noshaba, T., Rizwan, A. N., Ateeq, U. R., Mohammed, K. A. K. *Performance Evaluation of Machine Learning-Based Channel Equalization Techniques: New Trends and Challenges*, 2022. Journal of Sensors, vol. 2022. Available: https://doi.org/10.1155/2022/2053086. (visited on 05/02/2022).

[2] Ye, H., Li, Y. G., Juang, F. B-H. *Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems,* 2017. IEEE. Available: https://arxiv.org/pdf/1708.08514.pdf. (visited on 05/02/2022).

[3] Honkala, M., Korpi, D., Huttunen, J. M.J. *DeepRX: Fully Convolutional Deep Learning Receiver,* 2021. Available: https://ieeexplore.ieee.org/document/9345504. (visited on 03/20/2022).

[4] Prasad, R. *OFDM for Wireless Communications Systems*, 2004. The Artech House Universal Personal Communications Series. ISBN: 978-0-89006-571-6.

[5] Dahlman, E., Parkvall, S., Skold, J. *4G LTELTE-Advanced for Mobile Broadband*, 2014. Second Edition. ISBN: 978-0-12419-997-2.

[6] Beard, C., Stallings, W. *Wireless Communication Networks and Systems, 2016.* Pearson Higher Education, Inc.

[7] Rouphael, T, J. *RF and digital signal processing for software-defined radio a multi-standard multi-mode approach*, 2009. ISBN: 978-0-7506-8210-7.

[8] Pun, M., Morelli, M., Kuo, C. C. J. *Multi-Carrier Techniques For Broadband Wireless Communications A Signal Processing Perspective*, 2007. Vol 3. ISBN: 978-1-86094-946-3.

[9] Scikit-learn. *Multi-layer perceptron documentation*. Available: https://scikit-learn.org/stable/modules/neural_networks_supervised.html. (visited on 05/02/2022)

[10] Kattan, A., Abdulla, R., Geem, Z. W. *Artifcial neural network training and software implementation techniques*, 2011. Hauppauge: Nova Science Publishers, Inc.

[11] Scikit-learn. *Underfitting vs. Overfitting documentation.* Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html. (visited on 05/02/2022)