Diplomarbeit

Evaluierung von Deep Learning U-Net Architekturen für die Vorhersage der Signalstärke in Mobilfunknetzen

ausgeführt zum Zwecke der Erlangung des akademischen Grads
Diplom-Ingenieur
eingereicht an der TU Wien, Fakultät für Elektrotechnik und Informationstechnik

---

Diploma Thesis

Evaluation of Deep Learning U-Net Architectures for Signal Strength Prediction in Cellular Networks

submitted in satisfaction of the requirements for the degree
Diplom-Ingenieur
of the TU Wien, Faculty of Electrical Engineering and Information Technology

Viet Dung Thomas Tran, BSc

Matr.Nr.: 01402108

Betreuung: Univ.Prof. Dipl.-Ing. Dr.techn. **Markus Rupp**
Senior Scientist Dipl.-Ing. Dr.techn. **Philipp Svoboda**
Projektass. Dipl.-Ing **Lukas Eller**, BSc
Institut für Telekommunikation
Forschungsbereich Wireless Communications
Technische Universität Wien
Karlsplatz 13, 1040 Wien, Österreich

Wien, im November 2024

Kurzfassung

Eine effektive Planung von Funknetzen erfordert genaue Ausbreitungsmodelle, die das komplexe Verhalten elektromagnetischer Wellen, einschließlich Streuung, Mehrwegeausbreitung und Interferenzen, berücksichtigen. Herkömmliche empirische Modelle können diese physikalischen Eigenschaften oft nicht richtig darstellen. Im Gegensatz dazu liefern deterministische Modelle wie Ray Tracing genauere Vorhersagen, sind jedoch mit hohen Rechenkosten verbunden und erfordern sehr detaillierte Beschreibungen der städtischen Gebiete. Um diesen Herausforderungen Herr zu werden, gewinnen datengetriebene Modelle, insbesondere UNets, zunehmend an Aufmerksamkeit.

In dieser Arbeit wird ein neuronales Netz namens UNet analysiert, das durch Lernen aus etablierten Verteilungen der empfangenen Signalstärken in städtischen Gebieten wichtige Erkenntnisse über die Wellenausbreitung gewinnt. Einmal trainiert, kann es die Signalstärke über große Gebiete mit bemerkenswerter Effizienz und Genauigkeit vorhersagen. Darüber hinaus macht die Fähigkeit des UNet, Signalstärkeverteilungen aus einer begrenzten Anzahl von Messungen zu interpolieren, es zu einem wichtigen Hilfsmittel für die Planung neuer und die Optimierung bestehender Funknetze.

Ein zentrales Ziel dieser Forschungsarbeit ist es, die Beziehung zwischen der Anzahl der Eingangsmessungen und der Vorhersagegenauigkeit zu untersuchen. Die Ergebnisse zeigen, dass UNet selbst mit relativ wenigen Messungen eine beeindruckende Vorhersagegenauigkeit erreicht, obwohl der Nutzen durch Hinzufügen weiterer Messungen abnimmt. Diese Erkenntnis unterstreicht die Notwendigkeit, die Vorhersagegenauigkeit mit dem Messaufwand in Einklang zu bringen.

Darüber hinaus zeigt die Untersuchung, dass UNet Messunsicherheiten wie Rauschen und Global Positioning System (GPS)-Ungenauigkeiten wirksam ausgleicht und diese Diskrepanzen teilweise kompensiert. Seine Robustheit unter realen Bedingungen macht UNet zu einer guten Wahl für die Vorhersage der Signalstärke in Städten. Zusammenfassend lässt sich sagen, dass UNet eine ressourceneffiziente Lösung bietet, die eine schnelle und genaue Vorhersage der Signalstärke gewährleistet und gleichzeitig das transformative Potenzial von Deep-Learning-Techniken zur Verbesserung der Funknetzplanung und -optimierung aufzeigt.

Abstract

Effective radio network planning requires accurate propagation models that account for complex behaviors of electromagnetic waves, including scattering, multipath propagation, and interference. Traditional empirical models often fail to present properly these physical properties. In contrast, deterministic models such as ray tracing provide more precise predictions, but they come with high computational costs and require highly detailed descriptions of urban environments. In response to these challenges, data-driven models, particularly UNets, are gaining increasing attention.

This thesis analyzes a UNet neural network, which acquires significant insights into wave propagation mechanics by learning from established distributions of received signal strengths in urban areas. Once trained, it can predict signal strength over large areas with remarkable efficiency and accuracy. Furthermore, the UNet's capability to interpolate signal strength distributions from a limited number of measurements makes it an essential tool for planning new networks and optimizing existing ones.

A central objective of this research is to investigate the relationship between the number of input measurements and prediction accuracy. The results indicate that UNet achieves impressive prediction accuracy even with relatively few measurements, although the benefit of adding more measurements diminishes. This finding emphasizes the necessity of balancing prediction accuracy with measurement effort.

Additionally, the study demonstrates that UNet effectively addresses measurement uncertainties, such as noise and GPS inaccuracies, partially compensating for these discrepancies. Its robustness in real-world conditions makes UNet a strong choice for urban signal strength prediction. In conclusion, UNet offers a resource-efficient solution that ensures rapid and accurate signal strength forecasting while showcasing the transformative potential of deep learning techniques in enhancing radio network planning and optimization.

Contents

Chapter 1

Introduction

1.1 Motivation

Radio network planning is essential to operate a cellular network efficiently [1]. A thorough understanding of signal strength distribution in an urban area for a given antenna setup helps the network operator to define coverage areas, mitigate interferences, and allocate power and resources [2]. Consequently, accurate models for predicting signal strength are essential.

However, calculating received signal strength in wireless communications faces many challenges. Radio waves obey Maxwell's equations, and the complexities introduced by urban structures, antenna modeling, and their physical properties make the computation of the signal strength very resource-consuming. Ray tracing simplifies the propagation model by assuming radio signals propagate as rays, each traveling in a particular direction. Further assumptions, i.e., harmonic, high frequency, and free space propagation, reduce the complexity compared to the Maxwell approach. The propagation medium attenuates each ray, and obstacles on their paths scatter, diffract, or reflect the waves in the ray tracing model [3]. Therefore, it requires a complex description of the environment and a lot of computing power.

Empirical models offer an alternative approach that addresses the computational intensity of deterministic models. These models simplify the prediction process by utilizing equations with a small number of parameters derived from measurements. However, a significant drawback of empirical models is their limited accuracy as they neglect complex propagation mechanics in the urban environment [4].

As a complement to the other models, data-driven models are powerful tools gaining more attention since they can produce accurate signal strength predictions while being computationally efficient with a low execution time [5]. A reinterpretation of the problem allows the expression of the Radio Map Estimation (RME) task as a regression problem [4], where the network such as a UNet generates a radio map using information about the environment and the antenna setup. The UNet acquires extensive knowledge of propagation mechanics through comprehensive learning from various radio maps across different urban environments and antenna settings. The model can make highly accurate predictions for various urban scenarios and antenna configurations. Due to these advantages, this thesis emphasizes UNets. Advancements in neural networks will continue, leading to the development of more precise models and the inclusion of additional features in the future.

1.2 Related Works

Understanding existing methods for RME highlights the significance of UNets. This section begins by introducing classical models, outlining their strengths and limitations. It then continues to neural network approaches, discussing the current state of the art. Finally, the section concludes with a summary of the contributions of this thesis.

1.2.1 Empirical and Deterministic Models

There are two categories of classical radio map estimation models: empirical and deterministic. Empirical models are based on measurements and typically provide less accurate predictions for general urban wireless networks. In contrast, deterministic models are founded on physical properties and can represent the real world with greater accuracy. However, they tend to require significantly more computational resources.

A well-known empirical model for predicting radio wave propagation is the Okumura-Hata model [6]. This model is based on statistical measurements conducted in Tokyo. The model's equation considers several factors, including the carrier frequency, which ranges from 150MHz to 1500MHz, the transmitter height, which varies between 30m and 200m, and the distance, which spans from 1km to 20km. It also includes a correction factor that accounts for the type of area in which the prediction is made, categorized as open, suburban, or urban. The model incorporates the urban environment solely through a correction factor without considering individual building structures. Another empirical model is the COST 231 model, which covers a frequency range from 800MHz to 2000MHz to accommodate the GSM frequency band [7]. However, compared to more straightforward methods used in this thesis, the Okumura-Hata and COST 231 models are deemed too imprecise. For Okumura-Hata model predictions, the root mean square error (RMSE) is approximately 12-15dB [4].

Deterministic models, like ray tracing, are more effective when the predictions have to be more accurate than with empirical models. The fundamental simplification in ray tracing is the assumption of a harmonic electromagnetic wave that follows straight propagation paths. Nonetheless, the need for a highly accurate three-dimensional representation of the urban environment persists to account for reflections, scattering, and diffraction. A ray is reflected by interacting with the interface of the obstacles. Reflection law and Fresnel's equations determine the ray's direction and intensity. Diffracting occurs when objects split the incoming ray into infinite outgoing rays. Scattering alters the propagation paths depending on the property of the surface of the obstacles [3]. This requirement and the computation of the propagation of each ray make ray tracing methods computationally expensive despite the simplification of the propagation model. Since this model generates predictions based on physical properties, authors in [5], [8], [9] have performed simulations using ray tracing to generate the dataset [9], on which this thesis relies.

1.2.2 Neural Network Approaches for RME

Deep-learning approaches overcome the limitations of deterministic and empirical models. While they involve substantial computational complexity during the training phase, they become computationally efficient and can make highly accurate predictions once trained. Crucial challenges associated with neural networks include generalization and handling nonlinear relationships [10]. Furthermore, the training process requires extensive training data.

Multiple data-driven approaches for RME utilize at least some knowledge from the radio map in the training dataset to predict signal strength. One example of many elaborated by Romero et al. in [2] is the linear parametric RME, which calculates the signal strength by summing the weighted received powers of individual transmitters [2]. The limitations of this structure include low accuracy in non-line-of-sight conditions and an inability to generalize to unfamiliar urban scenarios and antenna placements.

Mentionable methods, including interpolation and Gaussian regression, can complete the radio map with sparse measurements of an area of interest [11]–[13]. These methods rely on measurement data availability, limiting their practicality for network planning tasks. This work aims for a moderate level of implementation complexity by utilizing the straightforward models provided by the `scipy` package, such as linear interpolation and k-nearest neighbors (KNN) regression. These methods form the foundation for evaluating the benchmark.

For example, neural networks used in image processing can predict radio maps even in regions where the radio network is underdeveloped. They achieve this by learning the average material and propagation properties from existing radio maps based on specific input parameters, including environmental information and antenna characteristics. UNets are a popular choice for RME, and the works in [4], [14], [15] emphasize the effectiveness. Their U-shaped structure is straightforward, and the overall network is computationally efficient. Hence, this thesis discusses this type of network in detail. The input parameters, which encompass both urban environments and antenna characteristics, can be described with a high degree of precision, e.g., by using seven parameters, i.e., surface type, height, and position information of buildings, the height of the terrain, the horizontal and vertical angle difference between the line of sight to the antenna, antenna height, carrier frequency, and antenna gain as referenced in [4]. Additionally, the authors in [15] define the radio network setup by incorporating a building model, an elevation map, the transmitter frequency, horizontal and vertical pointing offsets relative to the antenna, and transmitter power.

There are two methods for a network to create a complete radio map from the information passed through it. One method is the UNet described in [15], which generates the signal strength prediction for each position individually in an urban scenario. A vital advantage of this network is that it can forecast signal strength without using a complete radio map of an area of interest for training purposes. UNets, as discussed in [4] and in this thesis, are designed for image translation tasks. They take input images, such as those depicting urban environments, and generate comprehensive signal strength maps. For training purposes, it is crucial to have a comprehensive set of complete signal strength maps for the areas of interest. Suppose a vast

amount of training data is accessible to the network. In that case, the network computes the complete radio map for a specific urban area much faster than calculates it point by point.

Alternative structures are also considerable choices for RME, such as RadioUNet, as described by [8]. This architecture consists of two UNets arranged in a W-shape. The input for the second UNet includes the same data as the first UNet and the output from the first UNet. Further networks set their focus on completing radio maps from sparse measurements as in [16], [17]. However, besides sparse measurement of the radio networks, these networks need to include urban information in their input, making them suitable for radio network planning. This thesis modifies this concept for network calibration to enhance prediction accuracy and enable the completion of the radio map from measurements.

1.2.3 Contribution

This thesis contributes to the neural network approach for RME as follows.

- A UNet that processes city and antenna information as images and optionally includes measurement to generate a signal strength prediction. This thesis also compares the model with simple alternatives such as linear interpolation and KNN regression.
- The input includes a measurement channel in addition to the environment and antenna description for network calibration. The more measurements passed through the network, the more prior knowledge about the signal strength distribution it gains, and the more accurate the prediction will be. When the prediction of a sample delivers unsatisfactory results, measurements assist the network to generate more accurate predictions.
- This thesis explores how the measurement channel impacts UNet. The use of this channel is not mandatory. The network can rely on the model for unmeasured areas if it does not receive any measurements.
- This study demonstrates the robustness of the presented model against measurement distortions by testing it with various types of noise and intensity levels. Because of its robustness, this model is more practical for real-world measurements compared to other methods, such as interpolation.

1.3 Outline

When the urban scenarios are represented as images, which are essentially two-dimensional arrays, information about the position can be encoded in the index of the arrays, and the element's value represents a physical quantity, such as the height of a building, signal strength outside buildings, distance to an antenna, or antenna height. This thesis presents and analyzes a neural network that processes input arrays representing the urban scenario, i.e., the placement and height of buildings on a map, the basic setup of an antenna, i.e., the location and height where the antenna is mounted, and later a measurement map and generate an output which is an array

with detailed information about the signal strength at each location of interest, which is the street area in the city map. Signal strength inside buildings is out of interest, so the network neglects these predictions. Fig. 1.1 sketches the general task of the neural network. Chapter 2 and 3 will introduce the UNet and the SegNET, where the UNet remains in further investigations. Chapter 4 introduces the measurement map to the network, representing simulated signal strength measurements across urban areas. Furthermore, this work compares the results with those of linear interpolation and KNN regression, where the measurements serve as supporting points and observations, respectively. This thesis mainly focuses on assessing prediction quality by altering the number of measurements and simulating real-world measurement conditions by incorporating measurement errors in Chapter 4 and 5.

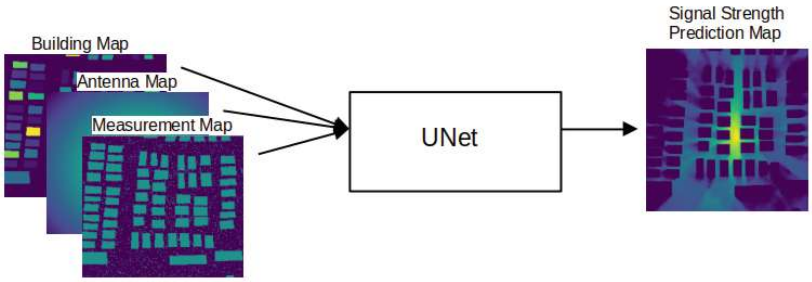


Fig. 1.1: Sketch of a Neural Network for RME

Chapter 2

Problem Statement

This work aims to design a neural network that processes input images that carry information about the urban scenario, antenna settings, and later measurements. An ideal candidate is a neural network based on convolutional operations, such as UNets or SegNETs. These networks have a simple structure consisting of convolutional and transposed convolutional layers. They are particularly effective for image processing because they require a relatively small number of parameters, making them computationally efficient. The total number of parameters depends on the architecture of the Convolutional Neural Network (CNN) and the chosen window, which is typically small.

2.1 Neural Networks

SegNET and UNet feature a similar U-shaped structure; however, the UNet includes connections between non-adjacent layers, known as skip connections. This work investigates both networks, utilizing SegNET to examine the skip connections in UNet.

2.1.1 SegNET

Initially, SegNETs, short for segmentation networks, were used for pixel-wise segmentation and classification tasks. Suitable design enables regression tasks as well. The architecture consists of an encoder network connected to a decoder network [18], [19]. Fig. 2.1 shows an exemplary SegNET. Typically, the encoder network consists of convolutional layers, and the decoder layer consists of convolutional or transpose convolutional layers, depending on the implementation. The layers are connected sequentially, i.e., after processing the input, the layer passes the output to the next layer. Suppose the network has n encoder and n decoder layers. The encoding process involves enumerating n layers from the input layer to the n-th layer, which is the final layer of the encoder. The encoder connects to the decoder by linking its last n layer to the n layer of the decoder. The layer connected to the n-th layer is the $n-1$-th layer[1]. The activation function usually concludes the operation of a layer, optionally followed by a pooling layer and a dropout layer, which decrease the size of the feature maps. In the case of a pooling layer, the max-pooling layer is a common choice. However, for pixel-wise classification or regression tasks,

[1]This is one way of enumerating the layers; other literature may use different enumerations.

the output image should have the same size as an input image. Therefore, methods to increase the image size are necessary, called up-sampling, at some point. Some papers, such as in [20], propose unpooling operation, while authors in [21] perform transpose convolution in their work. A combination of both methods is also possible. This thesis only uses transpose convolution and striding to restore the original image size. When using pooling layers, the decoder performs unpooling operations, and as in [19]. The i-th encoder layer passes the pooling indices to the unpooling layer. The pooling operation reduces the size of the feature maps. In contrast, the unpooling operation increases the size of the feature maps depending on the pooling indices it receives from the encoder.

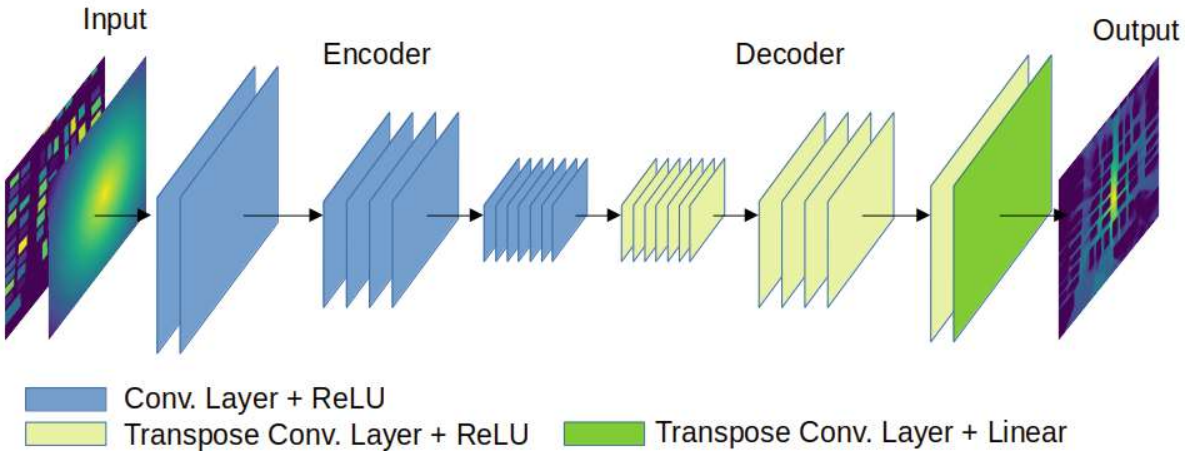


Fig. 2.1: Exemplary architecture of a SegNET.

2.1.2 UNet

UNets can be considered an evolution of SegNET, initially designed for segmentation tasks. Like SegNET, a UNet consists of an encoder consisting of convolutional layers and a decoder consisting of (transposed) convolutional layers, each usually terminated with an activation function. The difference between SegNET and UNet is the connection between the corresponding encoder and decoder layers, which is called the skip connection. Consider a UNet, where the encoder and the decoder have each n layers. The enumeration is the same as that for SegNET. An i-th decoder layer receives feature maps from an i-th encoder layer through a skip connection. It concatenates them with feature maps from a layer one level below, i.e., the $i-1$-th decoder layer, to form an input tensor for the (transpose) convolution operation of the i-th decoder layer. Unless it is the n-th decoder layer, the input of this layer is solely connected with the output of the n-th encoder layer. Pooling or unpooling at the end of each layer alters the size of the feature map. Alternatively, using striding is a valid option or a combination of both methods. An example of a typical UNet architecture based on [22] is shown in Fig. 2.2.

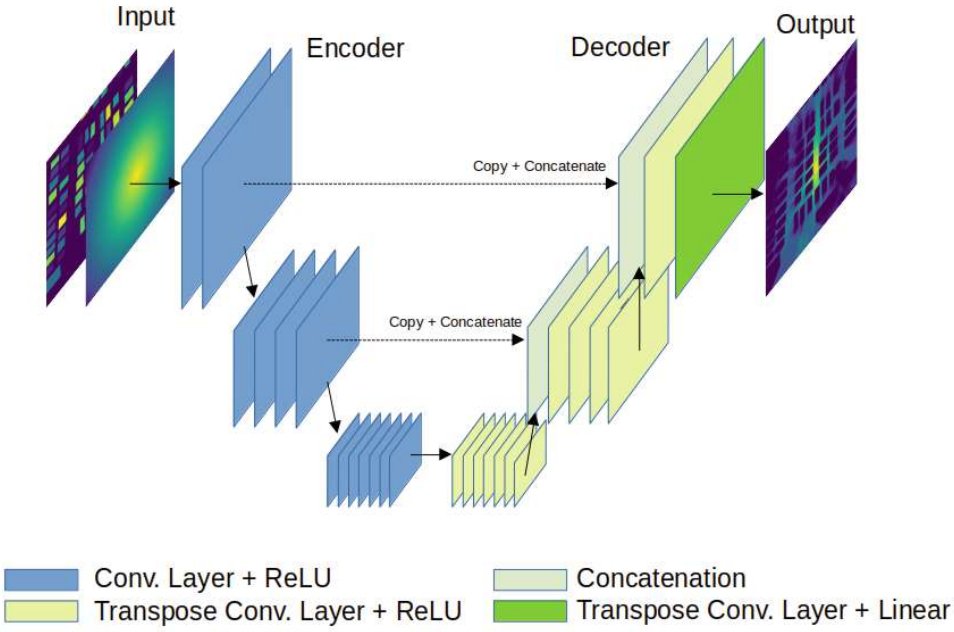


Fig. 2.2: Exemplary architecture of a UNet.

2.2 Convolutional and Transpose Convolution Operation

The main task of a (transpose) convolutional layer is performing (transpose) convolutional operations for image manipulation. In machine learning, a convolutional operation generates an output by a weighted sum of some input elements and a bias. Which input elements are selected depends primarily on the convolutional operation's window size and the output element index of the output array. Assuming a 2D convolution with a kernel $\boldsymbol{W}$ of size $(W_1 \times W_2)$, then the output element $\hat{\boldsymbol{Y}}_{kl}$ is given by

$$\hat{\boldsymbol{Y}}_{kl} = \sum_{i=1}^{W_1} \sum_{j=1}^{W_2} \boldsymbol{X}_{k+i\,l+j} \boldsymbol{W}_{ij} + \boldsymbol{B}_{kl}, \tag{2.1}$$

where $\boldsymbol{X}_{kl}$ is the k,l-th element of the input $\boldsymbol{X}$ and $\boldsymbol{B}_{kl}$ is the bias [23]. Unfortunately, the operation reduces the output size by the convolutional operation inherently by $W_1 - 1 \times W_2 - 1$. Therefore, additional processes, such as padding, must be performed to preserve the image size. The following Fig. 2.3 gives a visualization of the process.

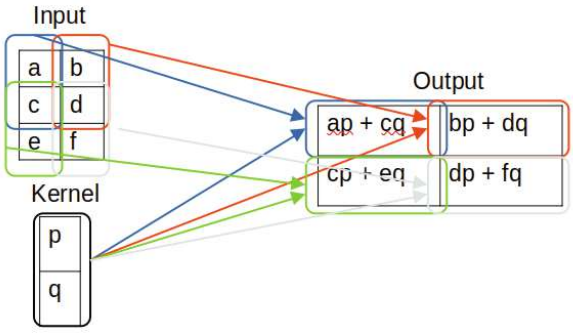


Fig. 2.3: Visualization of a convolution process for an input of size 2×3 and a window size of 1×2 [23].

A 2D transpose convolution performs as follows. Based on [24], assume a 2D input $\mathbf{X}$ of size $N_1 \times N_2$ with elements denoted with x_{ij} and a kernel with window size $W_1 \times W_2$ with elements w_{kl}. Contrary to convolutional operation, a transpose convolutional operation first generates for each element x_{ij} an intermediate array $\tilde{\boldsymbol{Y}}^{ij}$ of size $(N_1 + W_1 - 1) \times (N_2 + W_2 - 1)$ and for $i = 1, 2, \ldots, N_1$ and $j = 1, 2, \ldots, N_2$, whereby the intermediate array element $\tilde{y}_{ij,mn}$, is constructed as followed

$$\tilde{\boldsymbol{Y}}^{ij}_{mn} = \begin{cases} \boldsymbol{X}_{ij}\boldsymbol{W}_{m-i-1\ n-j-1} & \text{for } i \leq m \leq i + W_1,\ j \leq n \leq j + W_2 \\ 0 & \text{else} \end{cases}. \tag{2.2}$$

Then the output $\boldsymbol{Y}$ of size $(N_1 + W_1 - 1) \times (N_2 + W_2 - 1)$ is obtained by the summation

$$\boldsymbol{Y} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \tilde{\boldsymbol{Y}}^{ij}. \tag{2.3}$$

The process is visualized in Fig. 2.4.

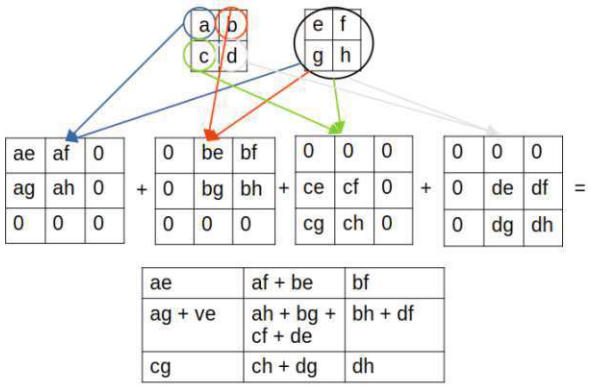


Fig. 2.4: A transpose convolution with an input of size 2×2 and a kernel with window size 2×2[24].

If an enlarged of the feature map is necessary, striding could be a considerable choice. A stride parameter of $p \times q$ forces the operation to generate intermediate arrays of size $(pN_1 + W_1 - p) \times (qN_2 + W_2 - q)$ and instead of moving the kernel by one element horizontally or vertically, the kernel will be moved by p elements horizontally or q elements vertically. Fig. 2.5 visualizes this operation.

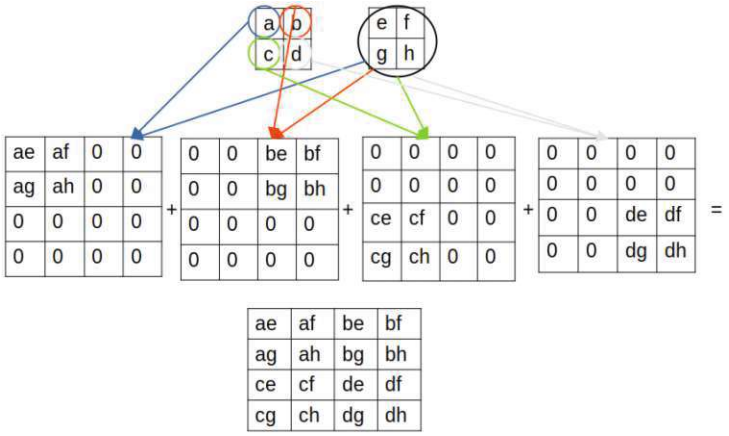


Fig. 2.5: A transpose convolution with an input of size 2×2 and a kernel with window size 2×2 and a stride of 2×2[24].

2.3 Training and Test Dataset

The paper [9] provides an extensive dataset. It contains 701 city maps of size 256×256 square meters from different cities. These maps are stored as two-dimensional image arrays of size 256×256, so each element of the image array corresponds to a spot on a map with a resolution of 1m^2. The value of the elements depends on the application area, such as height or signal strength.

These maps provide information about the location of buildings on the map and their heights. The range of heights starts from 6.6 meters to 19.8 meters. In this dataset, the tallest building defines the normalization, i.e., the tallest buildings have a value of 1. This simulation neglects ground elevation, meaning the entire street area has a value of 0.

Yapar et al. have done path loss simulation, using the ray-tracing software WinProp from Altair, for 80 different antenna locations for each city map in [9]. For each city map, there is one antenna mounted 3m above the roof of a building with a minimum height of 16.5m. A receiver height of 1.5m determines the path loss over the entire street area. In [9] path loss is defined as

$$P_L = (P_{\text{Rx}})_{\text{dB}} - (P_{\text{Tx}})_{\text{dB}}. \tag{2.4}$$

Usually $(P_{\text{Tx}})_{\text{dB}} > (P_{\text{Rx}})_{\text{dB}}$, resulting in negative path loss values, i.e. $P_L < 0$. The path loss range is $[-104\text{dB}, -75\text{dB}]$, i.e., the received power is at least -75dB less than the transmitted power. According to [9] the transmit power is $(P_{\text{Tx}})_{\text{dB}} = 23\text{dBm}$, so the range of receive power is $(P_{\text{Rx}})_{\text{dB}} \in [-81\text{dBm}, -52\text{dBm}]$. The path loss threshold is -104dB, determining the lowest

possible path loss value. The path loss map contains path loss over road areas that are normalized from 0 to 1, i.e., a value of zero means a path loss of $P_L = -104\text{dB}$ and one means $P_L = -75\text{dB}$. Due to the normalization, the path loss map is equivalent to the signal strength map, where zero corresponds to $(P_{\text{Rx}})_{\text{dB}} = -81\text{dBm}$, while one corresponds to $(P_{\text{Rx}})_{\text{dB}} = -52\text{dBm}$. Therefore, this thesis treats the normalized path loss map as the signal strength map. In addition, the signal strength is easier to access from the measurement because it does not require knowledge of the transmit power, and the device does not need to convert from receive power to path loss.

Tab. 2.1 summarizes the parameters of the simulations, provided by [9].

Tab. 2.1: Simulation parameters of the dataset [9]

Parameters	Value
Map size	256×256 pixels
Pixel length	1m
Rx height	1.5m
Transmit power	23dBm
Antenna type	Isotropic
Height range of the buildings	6.6m – 19.8m
Tx height	3m above rooftop
Center carrier frequency	3GHz
Channel bandwidth	20MHz
Path loss thresh hold	-104dB
Path loss range	29dB
Noise Figure	20dB

The antenna's position is stored as an array representing an area of size $256 \times 256\text{m}^2$ for each simulation. This array has a non-zero element, while all other elements are zero. This non-zero element carries the information about the antenna. The position can be retrieved by its index and the height by its value. Since the antenna installation is 3m above a building that is at least 16.5m tall, the tallest building in the dataset is 19.8m tall, the range of the vertical position of the antenna is $h_{\text{antenna}} \in [19.5\text{m}, 22.8\text{m}]$. The value is normalized, i.e., 1 for antennas at 22.8m height.

This thesis mainly used normalized maps, such as maps that provide the placement and height of buildings in an urban scenario, called building information maps. It also processes signal strength maps, which provide the antenna position, called antenna information maps. Other information that did not pass through the CNN is transmit power, carrier frequency, and bandwidth since the CNN is designed for signal strength prediction for a fixed frequency and bandwidth.

Since the dataset contains 701 city maps and each map has 80 simulations with different antenna locations, it contains 56080 simulations. This number of simulations was too much for this analysis, so a fraction served this thesis. The complete set would consume too much

memory[2]. First, 600 (because of the round number) of the 701 city maps were randomly selected. Five simulations with different antenna locations were chosen for each map, resulting in 3000 samples assigned to the training dataset. One hundred additional city maps, randomly selected and not previously included in the training data set, were utilized to create five simulations each, resulting in 500 samples designated for the test data set. By choosing this option, the urban maps assigned in the test dataset are unknown to the CNN. The training dataset does not overlap with the test dataset, i.e., no sample exists in both sets. The training and test data set can be divided into input and ground truth. Due to the image's resolution, the antenna coordinates correspond to the indices $[\hat{k}, \hat{l}]$, which can be expressed as a vector $\hat{\mathbf{x}}_{\text{antenna}} = [\hat{k}, \hat{l}]$ of the element that carries the antenna information in meters, and the height is its value, called h_{antenna}. Additional information, such as the distance from any position to the antenna, may help the network. Thus a two-dimensional array $\boldsymbol{X}_{\text{antenna}}$ is introduced, where an element with index k, l given by

$$\boldsymbol{X}_{kl,\text{antenna}} = 1 - \frac{\left\|\hat{\mathbf{x}}_{\text{antenna}} - [k,l]\right\|}{256\sqrt{2}} + h_{\text{antenna}}. \tag{2.5}$$

The distance from any point given by the vector $[k, l]$ to the antenna $\left\|\hat{\mathbf{x}}_{\text{antenna}} - [k,l]\right\|$ is first normalized to the largest possible and straightest distance in a map representing an urban area of $256 \times 256\text{m}^2$, which is a diagonal of length $d = 256\sqrt{2}$. Then, the value one is subtracted by these values, and normalized antenna height h_{antenna} is added. This results in elements corresponding to points closer to the antenna having larger values, while points further away from the antenna have smaller values.

The signal strength maps corresponding to their inputs are assigned to the ground truth of the training and test datasets, respectively. An example is shown in Fig. 2.6. The first and last images are building information and ground truth. They are taken directly from the dataset [9], denoted as $\boldsymbol{X}_{\text{building}}$ and $\boldsymbol{Y}$, and the middle image is the modified antenna information map $\boldsymbol{X}_{\text{antenna}}$. A conversion of the normalized signal strength to a dB scale enables a direct link to the received power in these plots. Signal strength inside buildings is out of interest, so the value is equal to the noise floor in the ground truth. Consideration of the signal strength inside buildings faces many challenges. The signal strength depends on the floor level since it varies with the distance to the antenna and on the internal architecture of the building. This case requires additional signal strength maps for each receiver's floor level and increases the measurement or simulation effort for the ground truth generation.

[2]Using 3500 samples occupies about 7.3GB of memory.

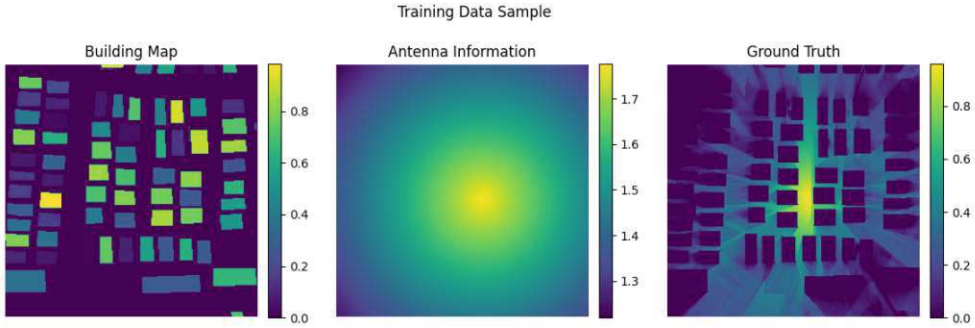


Fig. 2.6: A sample contains a building information map that represents the localization of facilities, an antenna information map, and the ground truth. This figure shows a training data sample

This particular training sample is an exemplary sample to demonstrate a single neural network prediction. To ensure a fair performance comparison, a test sample with characteristics similar to those of the training sample in Fig. 2.6, specifically the street share of the urban scenario, is necessary. This characteristic is computed by

$$r_{\text{street}} = \frac{|\mathcal{N}_{i,\text{street}}|}{256^2}, \tag{2.6}$$

where $|\mathcal{N}_{i,\text{street}}|$ is the number of pixels out of 256×256 pixels of the map that correspond to the street for the sample i. For the sample in Fig. 2.6 it is $r_{\text{street}} = 0.6637$. Therefore, a test sample, shown in Fig. 2.7 exhibits a similar street share of $r_{\text{street}} = 0.6684$.

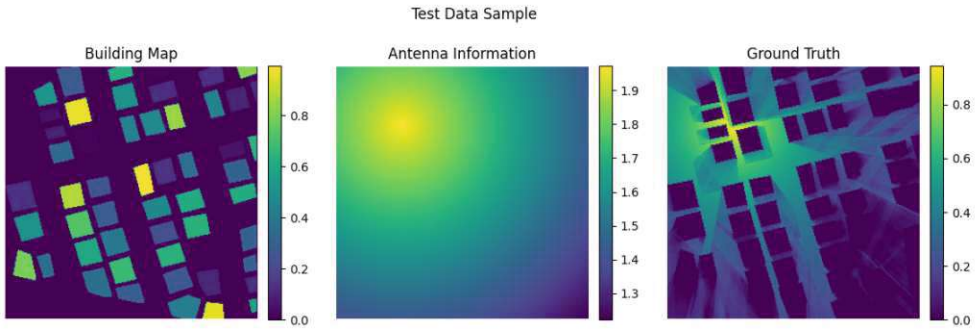


Fig. 2.7: Test data sample that contains building information map, antenna information map, and ground truth

2.4 Network Training

A neural network consists of a concatenation of (transpose) convolutional layers. It processes a three-dimensional input tensor $\boldsymbol{X}$ of size $N_1 \times N_2 \times K$, where $N_1 \times N_2$ is the spatial dimension, i.e., the size of the image in pixels, and K is the feature map dimension [25]. After a convolution layer, it is widespread to use a pooling layer, as in [25] or in [26]. Striding can also be used instead, as in [27], which has been done in this thesis since stride is an integrated feature of the (transpose) convolutional layer of `tensorflow`. The output $\hat{\boldsymbol{Y}}$ of the dimension $M_1 \times M_2 \times L$ ($M_1 \times M_2$ is the spatial dimension and L is the feature map dimension) of the network can be given by the function

$$\hat{\boldsymbol{Y}} = f(\boldsymbol{X}; \mathbf{w}), \tag{2.7}$$

where $\mathbf{w}$ are the parameters of the function $f(\cdot) : \mathbb{R}^{N_1 \times N_2 \times K} \rightarrow \mathbb{R}^{M_1 \times M_2 \times L}$. In supervised learning, the samples include the ground truth $\boldsymbol{Y}$, allowing a definition of a cost or loss function $\mathcal{L}(\cdot)$. The goal is to strive for optimal parameters $\mathbf{w}_{\text{opt}}$, by minimizing this cost or loss function, i.e.

$$\mathbf{w}_{\text{opt}} = \arg\min_{\mathbf{w}} \mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \arg\min_{\mathbf{w}} \mathcal{L}(\boldsymbol{Y}, f(\boldsymbol{X}; \mathbf{w})). \tag{2.8}$$

In most cases, finding a global minimum is impossible. Therefore, a technique called *Gradient Descent* is applied. The process starts with randomly initialized $\mathbf{w}_{k=0}$. Then at step k the parameters $\mathbf{w}_k$ are given by

$$\mathbf{w}_k = \mathbf{w}_{k-1} - \eta \frac{\partial \mathcal{L}(\boldsymbol{Y}, f(\boldsymbol{X}; \mathbf{w}))}{\partial \mathbf{w}}\bigg|_{\mathbf{w}=\mathbf{w}_{k-1}}, \tag{2.9}$$

where $\eta > 0$ is the step size or learning rate [28], [29]. The parameter *epoch* gives the number of iterations. The loss function has to be a convex function [30], and in the regression task, it is reasonable to choose it to be the averaged Mean Squared Error (MSE) since it takes the error of each element of the prediction into account. A classical approach involves calculating the MSE for each prediction $\hat{\boldsymbol{Y}}_n$ of the n-th sample in a dataset with N samples and then averaging all the MSE values., i.e.

$$\mathcal{L}(\{\boldsymbol{Y}_n\}_{n=1}^N, \{\hat{\boldsymbol{Y}}_n\}_{n=1}^N) = \frac{1}{N} \sum_{n=1}^{N} \|\boldsymbol{Y}_n - \hat{\boldsymbol{Y}}_n\|, \tag{2.10}$$

where $\boldsymbol{Y}_n$ is the ground truth of the n-th sample[3].

For large data set computing (2.10) can result in a large training duration. Stochastic gradient descent offers a more efficient solution to address this issue. At every iteration, it executes a gradient descent algorithm with a randomly chosen sample $\boldsymbol{X}_n$. The following equation represents its MSE loss

$$\mathcal{L}(\{\boldsymbol{Y}_n\}_{n=1}^N, \{\hat{\boldsymbol{Y}}_n\}_{n=1}^N) = \|\boldsymbol{Y}_n - \hat{\boldsymbol{Y}}_n\|. \tag{2.11}$$

[3]The n-th two-dimensional array from a set is indexed by one index, while the kl-th element of a two-dimensional array is indexed by two indices. If the two-dimensional array is not indexed, then is a generic sample, prediction or ground truth.

The loss function is estimated by randomly choosing the sample, and its complexity is strongly reduced compared to (2.10) [29]. Another algorithm used in this thesis is the Mini-Batch algorithm. Here, $1 < N_{\text{Batch}} < N$ samples $\boldsymbol{X}_n$ are selected randomly to compute the loss function. Let the set $\mathcal{X}_k$ contains randomly selected $\boldsymbol{X}_n$ and has the cardinality $|\mathcal{X}_k| = N_{\text{Batch}}$, then the MSE loss function is

$$\mathcal{L}(\{\boldsymbol{Y}_n\}_{n=1}^N, \{\hat{\boldsymbol{Y}}_n\}_{n=1}^N) = \frac{1}{N_{\text{Batch}}} \sum_{\boldsymbol{X}_n \in \mathcal{X}_k} \|\boldsymbol{Y}_n - \hat{\boldsymbol{Y}}_n\|. \tag{2.12}$$

The advantage of this method over stochastic gradient descent is the reduction of the variance of the gradient [30].

2.5 Baselines

This thesis mainly utilizes baselines for benchmarking neural networks. Multiple methods exist for generating them, including interpolation and regression techniques. The preferred choices were linear interpolation and KNN regression due to their simplicity of implementation. These methods only require knowledge of signal strength at specific locations. This information, which includes the position and corresponding signal strength, is the basis for completing a signal strength map. This knowledge contains the position and the signal strength of the position and forms supporting points for linear interpolation and observation for KNN regression.

2.5.1 Linear Interpolation

Various interpolations exist, including Lagrangian, Newton, and Hermite splines. All of them have the property $p(\mathbf{x}_i) = y_i$, where the pairs $(\mathbf{x}_i, y_i)_{i=0}^{N-1}$ are supporting points for an interpolation [31] (Note that N is not the number of samples in a data set, but the size of the set of supporting points). This analysis exploits piece-wise linear interpolation to generate the baselines. The information regarding the signal strength y_i with $i = 0, \dots N$ at specific positions $x_i \in \mathbb{R}$ provides the foundation for the linear interpolation. Let $\mathbf{x}_j$ and $\mathbf{x}_k$ denote the closest points to $\mathbf{x}_i$, i.e. there are no other points $\mathbf{x}_l$ such that $l \neq i, j, k$ with $|\mathbf{x}_i - \mathbf{x}_l| < |\mathbf{x}_i - \mathbf{x}_j|$ or $|\mathbf{x}_i - \mathbf{x}_l| < |\mathbf{x}_i - \mathbf{x}_k|$. With the corresponding values y_i, y_j, and y_k at these positions, the linear interpolation $p(\mathbf{x})$ determines a baseline within the triangle formed by these points. Using the linear interpolation

$$p(\mathbf{x}) = a_0 + a_1 x_1 + a_2 x_2, \tag{2.13}$$

where x_1, x_2 are the elements of $\mathbf{x}$ and $\mathbf{x}$ is inside the domain defined by the triangle, then the parameters a_0, a_1, a_2 can be obtained by solving the following set of equations

$$\begin{aligned} p(\mathbf{x}_i) &= y_i, \\ p(\mathbf{x}_j) &= y_j, \\ p(\mathbf{x}_k) &= y_k. \end{aligned} \tag{2.14}$$

2.5.2 KNN Regression

A common regression task is predicting out of observed data. Unlike interpolation, the values of the regression at the support point are primarily not equal to the actual value, i.e., $r(\mathbf{x}_i) \neq y_i = y(\mathbf{x}_i)$. The tuple $(\mathbf{x}_i, y_i)$ is called an observation in some literature such as in [32] or data point as in [33].

The computation of the signal strength $r(\mathbf{x})$ considers the k nearest neighbors $\mathbf{x}_j$ belonging to the set $\mathcal{X}_{\mathrm{NN},\mathbf{x}}$. These measurement positions are the closest to $\mathbf{x}$ and they form the set of nearest neighbors $\mathcal{X}_{\mathrm{NN},\mathbf{x}}$. Then, the following equation provides the KNN regression for these points

$$r(\mathbf{x}) = \frac{1}{k} \sum_{\mathbf{x}_j \in \mathcal{X}_{\mathrm{NN},\mathbf{x}}} y(\mathbf{x}_j). \tag{2.15}$$

2.6 Quality Measures for Evaluation

This section presents two measures to quantify the performance of the network. These are a modified RMSE, which quantifies the prediction accuracy solely for signal strength, and the R2 score, which gives some insight into the model.

2.6.1 Error Quantification

This will quantify the prediction error with (root) mean squared error ((R)MSE), which is defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{N_1 N_2} \sum_{k=1}^{N_1} \sum_{l=1}^{N_2} (\boldsymbol{Y}_{kl} - \hat{\boldsymbol{Y}}_{kl})^2}, \tag{2.16}$$

where $\boldsymbol{Y}_{kl}$ is the ground truth value, $\hat{\boldsymbol{Y}}_{kl}$ is the prediction value, and N is the array size of the prediction output or ground truth. First, collecting all possible two-dimensional indices results in a set of index tuples $\mathcal{N}_j = \{[0,0],[0,1],[1,0],\ldots,[N_1,N_2]\}$. (Note that N_1 and N_2 here are the dimensions of a single feature array) for a sample j from a dataset. City maps consist of building areas and street areas. Therefore, two disjoint subsets $\mathcal{N}_{j,\mathrm{street}}$ and $\mathcal{N}_{j,\mathrm{building}}$ together form the set $\mathcal{N}_j$, i.e. $\mathcal{N}_{j,\mathrm{street}} \cap \mathcal{N}_{j,\mathrm{building}} = \emptyset$, with $\mathcal{N}_j = \mathcal{N}_{j,\mathrm{street}} \cup \mathcal{N}_{j,\mathrm{building}}$, where $\mathcal{N}_{j,\mathrm{street}}$ contains all index tuple $[k,l]$ of sample j that are street areas and $\mathcal{N}_{j,\mathrm{building}}$ contains all index tuple $[k,l]$ of sample j that are building areas. For samples with larger building areas, the subset $\mathcal{N}_{j,\mathrm{building}}$ is usually larger than for samples with larger street areas, i.e. for the cardinality of these set it holds $|\mathcal{N}_{j,\mathrm{street}}| < |\mathcal{N}_{j,\mathrm{building}}|$. Signal strength inside buildings is out of interest. Therefore, the ground truth is $\boldsymbol{Y}_{kl} = 0$ for $[k,l] \in \mathcal{N}_{j,\mathrm{building}}$ and samples with a higher density of buildings tend to have more zero elements. Furthermore, the network is capable of accurately detecting building areas. A prediction for a sample j will have elements that are mostly close to zero for building areas, i.e., $\hat{\boldsymbol{Y}}_{kl} \approx 0$ for $[k,l] \in \mathcal{N}_{j,\mathrm{building}}$. In samples with high building densities, the subset $\mathcal{N}_{j,\mathrm{building}}$ is usually larger. In these cases, the ground truth contains many zero elements, while the predictions also have many close-to-zero values. According to (2.16) elements $[k,l] \in \mathcal{N}_{j,\mathrm{building}}$ will have a small contribution to the RMSE, since $\hat{\boldsymbol{Y}}_{kl} \approx \boldsymbol{Y}_{kl} = 0$ for $\mathcal{N}_{j,\mathrm{building}}$.

Consider a scenario where a network generates predictions for two samples with different building densities, and using (2.16) would deliver similar RMSE. These similar RMSEs can lead to the misinterpretation that the network can predict radio maps for samples representing densely developed regions with the same accuracy as for samples representing regions with few buildings. However, a sparsely developed area has a larger area where the prediction might deviate from the ground truth. Consequently, the deviation of each $\hat{\boldsymbol{Y}}_{kl}$ of the predictions belonging to the road area of the sparsely built-up area is more likely to be smaller than those in the densely built-up area sample. Computing the RMSE only for predicted values in street areas mitigates this effect. Let $|\mathcal{N}_{j,\text{street}}|$ be the number of elements in $\mathcal{N}_{j,\text{street}}$ or the number of elements belonging to street areas. Then, the RMSE for predicting only street areas is

$$\text{RMSE}_{\text{street}} = \sqrt{\frac{1}{|\mathcal{N}_{j,\text{street}}|} \sum_{[k,l]\in\mathcal{N}_{j,\text{street}}} (\boldsymbol{Y}_{kl} - \hat{\boldsymbol{Y}}_{kl})^2}. \tag{2.17}$$

2.6.2 R2 Score

This thesis determines the quality of the model using the coefficient of determination, commonly referred to as the R2 score. The R2 score measures how well the independent variables explain the variation in the model. It also provides a quality indicator for the fit [34], [35]. This analysis uses the R2 score method from `sklearn` package, which computes the R2 score according to the following equation

$$R2 = 1 - \frac{\sum_{k,l}(\boldsymbol{Y}_{kl} - \hat{\boldsymbol{Y}}_{kl})^2}{\sum_{k,l}(\boldsymbol{Y}_{kl} - \bar{\boldsymbol{Y}})^2}. \tag{2.18}$$

Equation (2.18) considers a single ground truth sample $\mathbf{Y}$ with elements $\boldsymbol{Y}_{kl}$ and a prediction $\hat{\mathbf{Y}}$ with elements $\hat{\boldsymbol{Y}}_{kl}$. The value $\bar{\boldsymbol{Y}}$ is the mean of all elements $\boldsymbol{Y}_{kl}$ in $\mathbf{Y}$ and is obtained by the following equation

$$\bar{\boldsymbol{Y}} = \frac{1}{N_{\text{sample}}} \sum_{k,l} \boldsymbol{Y}_{kl}, \tag{2.19}$$

where N_{sample} is the number of elements in $\mathbf{Y}$. The variation is given by $(\boldsymbol{Y}_{kl} - \bar{\boldsymbol{Y}})^2$, while $(\boldsymbol{Y}_{kl} - \hat{\boldsymbol{Y}}_{kl})^2$ is the squared error.

An R2 score of R2 = 1 would result from $\hat{\boldsymbol{Y}}_{kl} = \boldsymbol{Y}_{kl}$, meaning that the prediction is equal to the ground truth, which is the best achievable score. In this case, the model predicts with the highest accuracy. If a model predicts every element equal to the mean of $\mathbf{Y}$, i.e., $\hat{\boldsymbol{Y}}_{kl} = \bar{\boldsymbol{Y}}$, $\forall i$, regardless of the input variable, then R2 score is R2 = 0. A prediction from a model with this R2 score is called an imperfect prediction according to [34]. An R2 score below indicates a model worse than the imperfect prediction. It can theoretically go down to R2 = $-\infty$, since the squared error $(\boldsymbol{Y}_{kl} - \hat{\boldsymbol{Y}}_{kl})^2$ can be arbitrarily large, i.e. the model can be arbitrarily bad.

Chapter 3

Neural Network Evaluation

This chapter examines the impact of skip connections on UNet's performance compared to SegNET's. The training of both networks took place using the same dataset to ensure a fair comparison. The resulting learning curves provide insights into convergence, loss, and steepness in the training process. Predictions of selected samples show practical and visual results. However, the averaged RMSE determines and assesses the overall performance of the networks, facilitating comparison. The R2 score enables the evaluation of the model.

3.1 SegNET

3.1.1 Network Architecture and Training

The input of the SegNET consists of a building information map and a modified antenna information map, as discussed earlier, i.e. $\boldsymbol{X} = (\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}})$ and the output is a single image output, which is the prediction and denote as $\hat{\boldsymbol{Y}}$. The CNN consists of four encoder and decoder layers in this simulation.

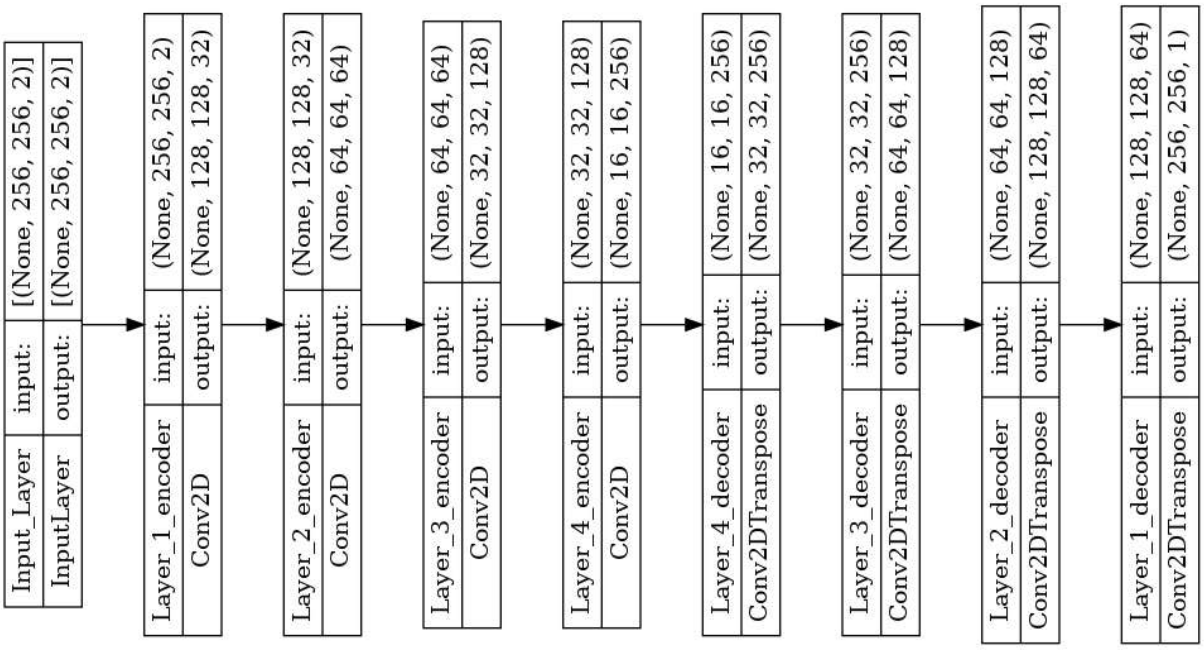


Fig. 3.1: Sketch of a SegNET with four encoder and four decoder layers

The architecture of the CNN is shown in figure 3.1 and is built up as follows.

- **Encoder**
  - `Input_Layer`
    - Input: Tensor of size $256 \times 256 \times 2$, consists of Building and Antenna Information, each of size 256×256
    - Output: Tensor of size $256 \times 256 \times 2$
    - Connected to `Layer_1_encoder`
  - `Layer_1_encoder`
    - Input: Tensor of size $256 \times 256 \times 2$
    - 2D Convolutional Layer, 32 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $128 \times 128 \times 32$
    - Connected to `Layer_2_encoder`
  - `Layer_2_encoder`
    - Input: Tensor of size $128 \times 128 \times 32$
    - 2D Convolutional Layer, 64 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $64 \times 64 \times 64$
    - Connected to `Layer_3_encoder`

  - `Layer_3_encoder`
    - Input: Tensor of size $64 \times 64 \times 64$
    - 2D Convolutional Layer, 128 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $32 \times 32 \times 128$
    - Connected to `Layer_4_encoder`
  - `Layer_4_encoder`
    - Input: Tensor of size $32 \times 32 \times 128$
    - 2D Convolutional Layer, 256 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $16 \times 16 \times 256$
    - Connected to `Layer_4_decoder`

- **Decoder**
  - `Layer_4_decoder`
    - Input: Tensor of size $16 \times 16 \times 256$
    - Transpose 2D Convolutional Layer, 256 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $32 \times 32 \times 256$
    - Connected to `Layer_3_decoder`
  - `Layer_3_decoder`
    - Input: Tensor of size $32 \times 32 \times 256$
    - Transpose 2D Convolutional Layer, 128 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $64 \times 64 \times 128$
    - Connected to `Layer_2_decoder`
  - `Layer_2_decoder`
    - Input: Tensor of size $64 \times 64 \times 128$
    - Transpose 2D Convolutional Layer, 64 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $128 \times 128 \times 64$
    - Connected to `Layer_1_decoder`
  - `Layer_1_decoder`
    - Input: Tensor of size $128 \times 128 \times 64$

- Transpose 2D Convolutional Layer, 1 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `Linear`
- Output: Tensor of size $256 \times 256 \times 1$

The goal of this work is different from the focus of finding an optimal network. So, the decision fell on this setup due to reasonable training complexity with reasonable computational resources and good prediction results. The network has a window size of (4×4) and a stride parameter of (2×2). Imagine that a two-dimensional array input can be interpreted as an object that can be traversed horizontally and vertically. After computing one output element of a feature map, the window moves vertically or horizontally by two elements. Furthermore, padding was used so that the size reduction only results from striding. For a single convolutional kernel and an array that represents a single image, i.e., an array of size $N \times N \times 1$, these choices result in an array with half the feature map size of the input, i.e., the size of the output array is $\lfloor \frac{N}{2} \rfloor \times \lfloor \frac{N}{2} \rfloor \times 1$. Since the choice is this method, a pooling layer can be omitted.

The transpose convolution operation with the same settings for `stride`, `window_size` and `padding` as for convolutional layer doubles the size of an image, i.e. for a single kernel and an input array of size $N \times N \times 1$ that contains a single image, the transpose convolution generates output arrays of size $2N \times 2N \times 1$. All layers have a `ReLU` activation, except the first decoder layer, which has a `linear` activation to satisfy the regression task requirement.

The training of the CNN took place using the training dataset, discussed in Section 2.3, using 200 epochs, a batch size of 16, and a default setting of the learning rate of 10^{-3}. The MSE serves as the loss function, which is given by

$$\text{MSE} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\boldsymbol{Y}_{ij} - \hat{\boldsymbol{Y}}_{ij})^2, \tag{3.1}$$

$\boldsymbol{Y}_{ij}$ corresponds to an element of the ground truth and $\hat{\boldsymbol{Y}}_{ij}$ to an element of the prediction, and $N^2 = 256^2$ is the number of elements in the ground truth array. The loss, computed with normalized values for $\boldsymbol{Y}_{ij}$ and $\hat{\boldsymbol{Y}}_{ij}$, requires a conversion to a dB scale for presentation, providing a direct relation to the power. According to [9] the following equation determines the conversion

$$\hat{\boldsymbol{Y}}_{\text{dBm}} = -104\text{dBm} + 29\text{dBm}\hat{\boldsymbol{Y}}_{\text{scaled}} + \underbrace{23\text{dBm}}_{\text{transmit power}}, \tag{3.2}$$

where $\hat{\boldsymbol{Y}}_{\text{scaled}}$ is the SegNET's prediction. The ground truth undergoes the same conversion. Consequently, the conversion of the normalized MSE loss to RMSE loss in dB scale is computed by

$$\text{MSE}_{\text{dB}} = 29\sqrt{\text{MSE}}. \tag{3.3}$$

Due to the complexity of implementation with the `tensorflow` package, loss exclusively for street areas was not applied. Fig. 3.2 shows the learning curve converted to the dB scale. The noisy behavior of the learning curves is present and can affect the final loss. The blue curve,

which corresponds to the RMSE of the training data, and the orange curve, which corresponds to the RMSE of the test data, diverge from each other. While the validation loss converges at about 50 epochs, the training loss decreases even for more significant epochs, increasing the gap between the training and test data loss, reaching about 1dB in this simulation. Consequently, the RMSE of the test data cannot be improved by increasing the number of epochs.

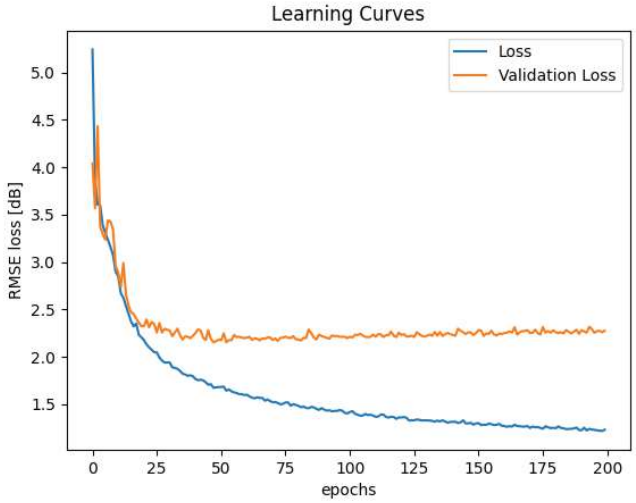


Fig. 3.2: Learning curves for SegNET. The CNN is trained with 3000 training samples, 200 epochs, and a batch size equal to 16. The RMSE loss for the training data is shown as the blue curve. Furthermore, 500 test data are used to compute the validation loss, which is the orange curve.

3.1.2 Prediction Results

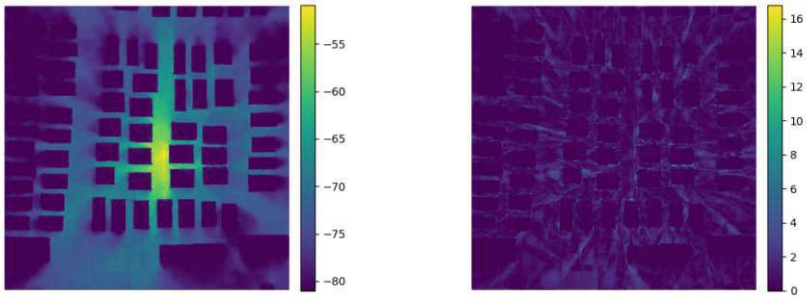
The prediction of a given training sample, shown in Fig. 2.6, and the one of a particular test sample, shown in Fig. 2.7, are presented. Since the activation of the output layer of the neural network is chosen as a linear activation, values outside the normalized range can be expected. For elements of the prediction that have values below zero, those elements are truncated to zero. This truncation is an integral part of the process for every prediction and baseline throughout this thesis. Then, converting the network's output to the dB scale remains the last step.

An introduction of an error map enables a visual evaluation of the quality of the prediction. This map highlights places where the prediction deviates from the ground truth. It is a two-dimensional array $\boldsymbol{E}$ with elements defined as

$$\boldsymbol{E}_{kl} = |\hat{\boldsymbol{Y}}_{kl,\mathrm{dBm}} - \boldsymbol{Y}_{kl,\mathrm{dBm}}|, \tag{3.4}$$

where $\hat{\boldsymbol{Y}}_{kl,\mathrm{dBm}}$ is a element of the prediction in dB and $\boldsymbol{Y}_{kl,\mathrm{dBm}}$ is a element of the ground truth in dB. According to its definition, this error map shows the absolute signal strength error at each point on the map. Introducing the RMSE from 2.17 helps to quantify the prediction quality.

The prediction of a training data sample shown in Fig. 2.6 is shown in Fig. 3.3a while the pixel-wise deviation from the ground truth is displayed in Fig. 3.3b. These figures mediate that the SegNET can detect the buildings and set all pixels belonging to buildings to values close to zero, which is about −81dBm, which can be interpreted as a segmentation task. Furthermore, the SegNET can predict the signal strength in street areas, so we can see that the network can perform regression tasks. Errors can be found near the antenna and mainly at the corner of the shadow.



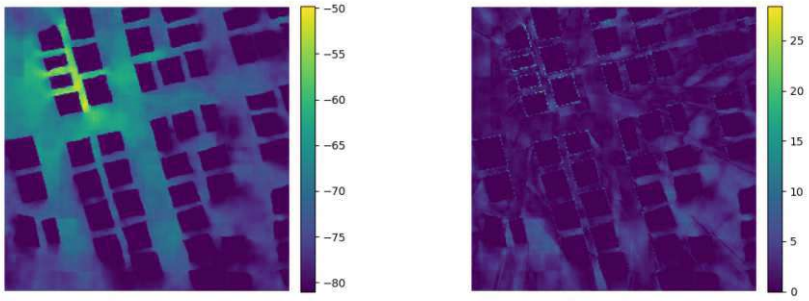
(a) prediction in dB scale **(b)** Absolute error in dB scale

Fig. 3.3: SegNET prediction of a training data sample and pixel-wise prediction error

The RMSE is

$$\boxed{\text{RMSE} = 1.1956\text{dB}.}$$

A particular test data sample's prediction, shown in Fig. 2.7, was analyzed. This sample has almost the same proportion of streets as the training sample above. Therefore, this sample is suitable for comparing the neural network's performance for unknown urban scenarios. The prediction and the error map are given in Fig. 3.4. As seen in the figure, the SegNET can determine the locations of the buildings. The error map reflects this ability, where building areas usually show low error. A blurry distribution of the signal strength around the antenna can be observed for prediction in street areas. Compared to the prediction of the training sample shown in Fig. 3.3, the details of the signal strength distribution, such as reflection, shadowing, diffraction, and other propagation mechanics, are less pronounced and get worse when the distance to the antenna is increased. The error map also shows these problems, indicated by a lighter color in Fig. 3.4b. Some pixels of the predictions can have errors of more than 25dB.



(a) prediction in dB scale **(b)** Absolute error in dB scale

Fig. 3.4: SegNET prediction of a training data sample and pixel-wise prediction error

It can be stated that the quality of the prediction is rather poor, which is reflected by the RMSE of

$$\boxed{\text{RMSE} = 2.8927\text{dB}.}$$

Since two predictions are not representative of determining the performance of the CNN, the averaged RMSE expresses the performance of the Network. The prediction accuracy depends on whether the sample belongs to the training or test data set. Therefore, this analysis investigates these cases separately. The first step is to compute the RMSEs of the predictions for all 3000 samples from the training data set, which results in a distribution of the RMSEs of all predictions from samples from the training dataset. This distribution provides the calculation of the averaged RMSE and variance. These values show the expected RMSE of the prediction of a sample from the training set and the variation of the RMSEs. The RMSE for a training sample is

$$\boxed{\text{RMSE} = (1.297 \pm 0.2066)\text{dB}.}$$

If the same is done for all 500 samples from the test data set, the following results can be obtained.

$$\boxed{\text{RMSE} = (2.4789 \pm 0.3416)\text{dB}}$$

The SegNET can predict training samples with higher accuracy, while the predictions of test data samples tend to be more erroneous. The higher accuracy of the training sample prediction is due to the training of the network since it has seen the input and the ground truth during the training process. The parameters can be optimally tuned to perform the regression tasks for training samples. The test samples are unknown to the network, so it must rely solely on the trained parameters to perform the prediction.

The R2 Score for training samples is computed by calculating the R2 Score of each sample from the training data set and averaged over all R2 Scores. This computation results in an averaged R2 for the training data set, which is

$$\boxed{\text{R2} = 0.8669}$$

and the same process for the test data set leads to

$$\boxed{\text{R2} = 0.64.}$$

The training data samples have a more significant R2 score than the test data, which means that the model can explain most of the variation of the training samples. It also implies that the regression model works modestly for the test data samples. However, predicting a test data sample can be a basis for determining a coarse region where significant signal strength is possible.

3.2 UNet

For a fair comparison between these networks, implementing a UNet consists of the same number of layers for the encoder and decoder. Furthermore, the layer settings (window size, stride, activation of a layer) correspond to the SegNET from the previous section. This section will analyze the impact of the skip connections on the predictions. Furthermore, the UNet is trained using the MSE Loss function with the same number of epochs = 200 and batch size = 16.

3.2.1 Network Architecture and Training

A UNet processes an input that consists of a building information map and antenna information map, i.e., $\boldsymbol{X} = (\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}})$ and generates a single prediction $\hat{\boldsymbol{Y}}$. This UNet contains a four-layer encoder and a four-layer decoder. Since the output of the i-th level decoder layer and the output layer of an $i+1$-th level encoder layer head to the decoder inputs of the $i+1$-th level layer, the network merges those feature maps before these maps enter the decoder layer. Ideally, these outputs have identical image sizes, e.g., the encoder output is of size $N_1 \times N_2 \times K_1$, where $N_1 \times N_2$ is the size of a single feature map and K_1 is the number of feature maps of the output. This output is merged with a feature map from a lower level decoder layer of size $N_1 \times N_2 \times K_2$ to form an input of size $N_1 \times N_2 \times (K_1 + K_2)$. However, this is not mandatory since the image size can be changed, e.g., by cropping. In this thesis, the selected parameters for each convolutional layer made cropping unnecessary. Due to the larger input tensors of the layers, the UNet has more trainable parameters than a SegNET with the same parameters (2 395 041 trainable parameters for SegNET and 2 723 233 trainable parameters for UNet). Furthermore, instead of pooling, this network reduces the image size using the stride method. The architecture is given in Fig.3.5.

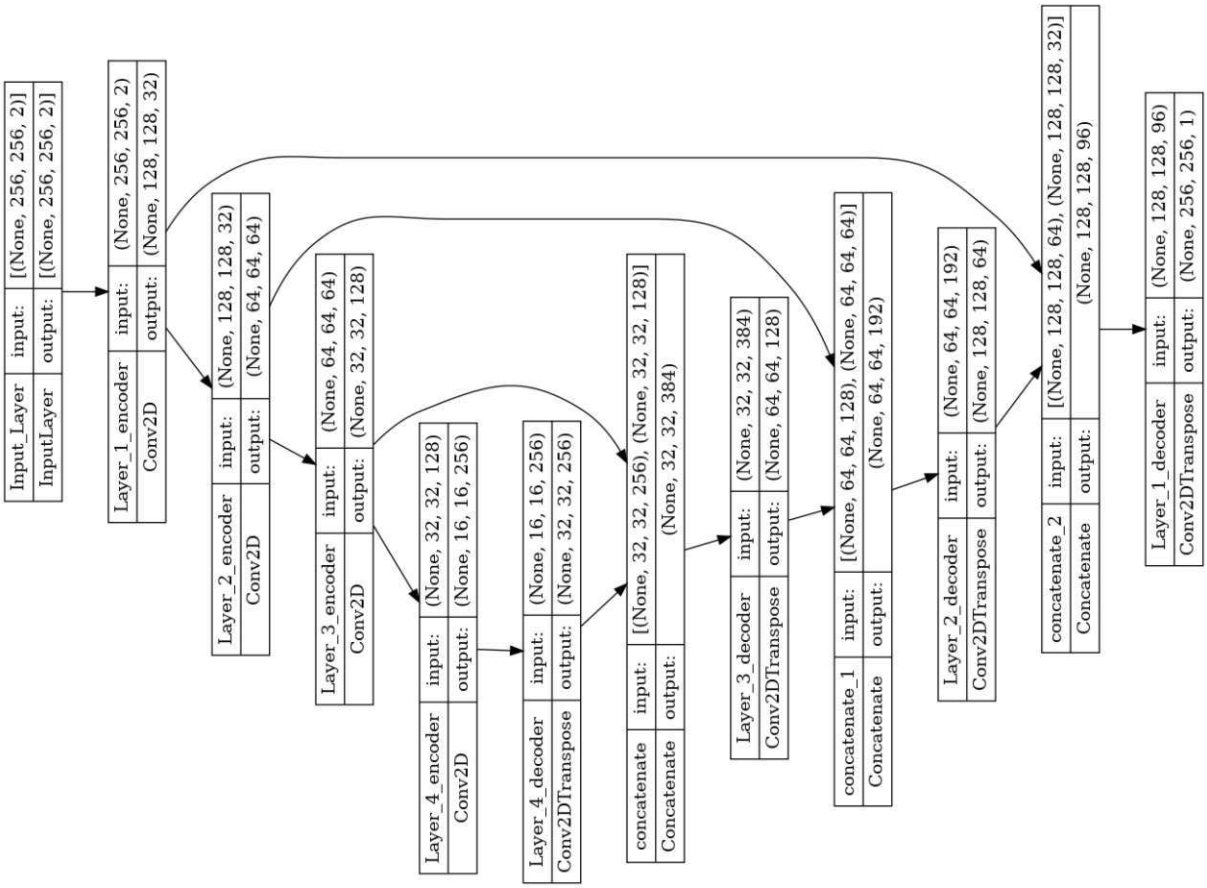


Fig. 3.5: Sketch of a UNet with four encoder and four decoder layers

Moreover, it is constructed with the following components.

- **Encoder**
  - `Input_Layer`
    - Input: Tensor of size $256 \times 256 \times 2$, consists of Building and Antenna Information, each of size 256×256
    - Output: Tensor of size $256 \times 256 \times 2$
    - Connected to `Layer_1_encoder`
  - `Layer_1_encoder`
    - Input: Tensor of size $256 \times 256 \times 2$
    - 2D Convolutional Layer, 32 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $128 \times 128 \times 32$
    - Connected to `Layer_2_encoder`, `Layer_1_decoder`
  - `Layer_2_encoder`

  - Input: Tensor of size $128 \times 128 \times 32$
  - 2D Convolutional Layer, 64 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
  - Output: Tensor of size $64 \times 64 \times 64$
  - Connected to `Layer_3_encoder`, `Layer_2_decoder`
- `Layer_3_encoder`
  - Input: Tensor of size $64 \times 64 \times 64$
  - 2D Convolutional Layer, 128 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
  - Output: Tensor of size $32 \times 32 \times 128$
  - Connected to `Layer_4_encoder`, `Layer_3_decoder`
- `Layer_4_encoder`
  - Input: Tensor of size $32 \times 32 \times 128$
  - 2D Convolutional Layer, 256 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
  - Output: Tensor of size $16 \times 16 \times 256$
  - Connected to `Layer_4_decoder`

- **Decoder**
  - `Layer_4_decoder`
    - Input: Tensor of size $16 \times 16 \times 256$
    - Transpose 2D Convolutional Layer, 256 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $32 \times 32 \times 256$
    - Connected to `Layer_3_decoder`
  - `Layer_3_decoder`
    - Input: Tensor of size $32 \times 32 \times 384$
    - Transpose 2D Convolutional Layer, 128 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
    - Output: Tensor of size $64 \times 64 \times 128$
    - Connected to `Layer_2_decoder`
  - `Layer_2_decoder`
    - Input: Tensor of size $64 \times 64 \times 192$

* Transpose 2D Convolutional Layer, 64 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `ReLU`
* Output: Tensor of size $128 \times 128 \times 64$
* Connected to `Layer_1_decoder`

– `Layer_1_decoder`

* Input: Tensor of size $128 \times 128 \times 96$
* Transpose 2D Convolutional Layer, 1 Kernel, Window size: 4×4, Stride: 2×2, Padding: `same`, Activation: `Linear`
* Output: Tensor of size $256 \times 256 \times 1$

After setting up a network, it underwent training with the same training data set used to train the SegNET using epochs = 200, batch size = 16, and a learning rate of 10^{-3}. Again, since normalized maps, i.e., building maps, antenna maps, and ground truth, were used to train the network, the results of the learning curves were converted to the dB scale when presented. The learning curves for the loss and validation loss of the UNet training are shown in Fig. 3.6. The learning curves of the SegNET training are also present in this figure as semitransparent lines for comparison. After 200 epochs, the training loss of the UNet is 1.1348dB, while the SegNET has 1.2318dB, which is an improvement of 0.097dB. However, the training curves suffer from a noisy behavior such that, by chance, the loss of the SegNET could outperform UNet after 200 epochs. For validation loss, the result is 2.2256dB for UNet compared to 2.2762dB for SegNET, which is a reduction of 0.0506dB. In Fig. 3.6, the validation learning curves of SegNET and UNet are similar, while it is evident from this figure that for training samples, the loss curve is slightly lower for a UNet.

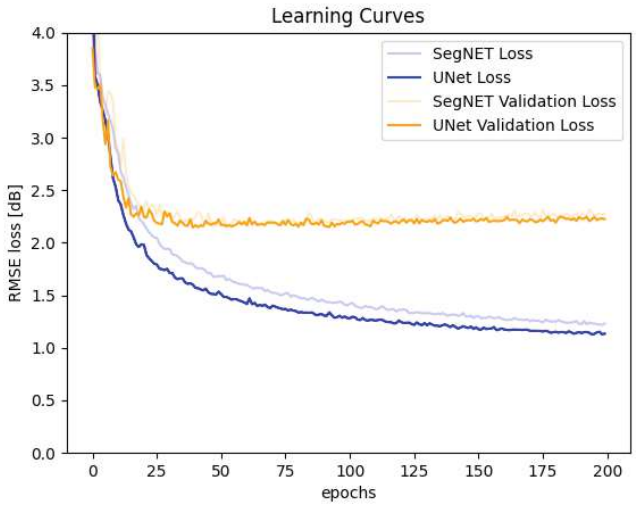
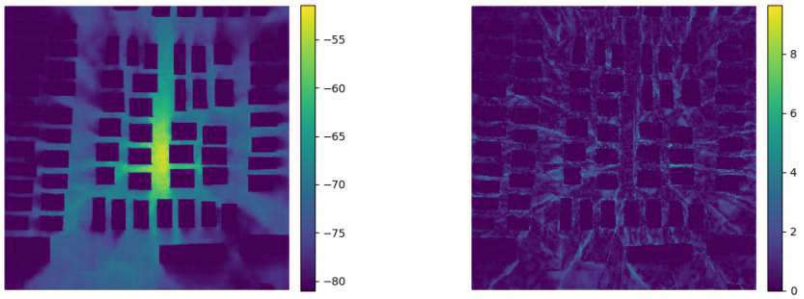


Fig. 3.6: Learning curves for UNet. These curves are shown in semitransparent color for comparison with the learning curves for SegNET.

3.2.2 Prediction Results

The prediction of a particular training sample, namely the sample shown in Fig. 2.6, is shown in Fig. 3.7a along with the corresponding error map in Fig. 3.7b. Since SegNETs can predict training samples with remarkable accuracy, a network with more trainable parameters should be able to predict with at least the same accuracy as the SegNET. Figure 3.7a shows reasonable illumination of signal power in street areas and some shadows cast by buildings. However, these shadows do not have sharp edges as in the ground truth, reflected in the deviation in these areas in the absolute error map in Fig. 3.7b. Furthermore, the illumination details become less pronounced for areas far from the antenna.



(a) prediction in dB scale **(b)** Absolute error in dB scale

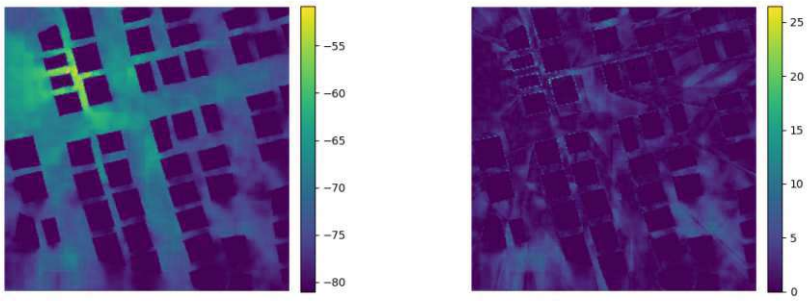
Fig. 3.7: UNet prediction of a training data sample and pixel-wise prediction error

For this prediction, the RMSE is

$$\boxed{\text{RMSE} = 1.0016\text{dB}.}$$

This result shows that the UNet predicts this sample more accurately than the SegNET, which generates a prediction for this particular training sample with an RMSE of 1.1956dB.

Fig. 3.8 shows the prediction for a test sample together with the corresponding error map. Similar to the SegNET prediction of this sample, the prediction shows a blurry receive power around the area. Coarse details of the signal strength distribution are recognizable mostly around the antenna, and they get lost when the position is further away from the antenna. The absolute error can reach more than 25dB for some pixels in the error map. Building areas can be detected accurately, showing the strength of UNets for segmentation tasks.



(a) prediction in dB scale **(b)** Absolute error in dB scale

Fig. 3.8: UNet prediction of a test data sample and pixel-wise prediction error

The RMSE of this prediction is

$$\boxed{\text{RMSE} = 2.8347\text{dB}.}$$

Moreover, it is slightly smaller than SegNET's prediction, which has an RMSE of 2.8927dB.

The averaged RMSE and variance can determine the overall performance of the UNET. Therefore, as in the previous section, the mean and variance of all RMSEs of all predictions of a set were calculated. The result for training samples is

$$\boxed{\text{RMSE} = (1.1883 \pm 0.1987)\text{dB}}$$

whereas for test samples, the result is

$$\boxed{\text{RMSE} = (2.4747 \pm 0.3268)\text{dB}.}$$

Compared to SegNET, where the RMSE is (1.297 ± 0.2066)dB for training samples and (2.4789 ± 0.3416)dB for test samples, the UNet exhibits a marginal increase in accuracy. However, this architecture will be used for further simulations and not perform any modification to the current network, except in Chapter 4, where the network includes an additional layer dedicated for measurements.

Predicting signal strength for complicated urban scenarios can be challenging for neural networks, and the signal strength distribution can become very complex. The signal has to deal with more obstacles, resulting in reflection or scattering. Furthermore, these objects can alter the signal's propagation path. If the network cannot handle these propagation mechanics properly, then the RMSE is expected to be significant. At the same time, a sparsely built area causes less scattering or shadowing. This urban environment may simplify the propagation mechanics and result in predictions with low RMSE. Therefore, it is interesting to analyze the training and test data samples where the UNet predicts the lowest and highest RMSE, respectively, and investigate if there are any correlations with building density.

Figure 3.9 shows a training sample and a test sample whose UNet predictions have the lowest RMSE. As can be seen from this figure, the placement of the buildings allows the signal to propagate with few obstacles in the vicinity of the transmitter.

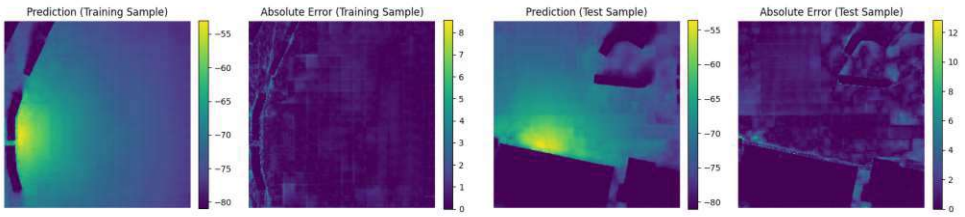


Fig. 3.9: Predictions of training and test sample that show the lowest RMSE

The results are shown in the following table.

	Training Sample	Test Sample
RMSE	0.6158dB	1.3538dB
r_{street}	0.9609	0.7246

Predictions for the training and test samples with the highest street share offer exciting insights. They could have generally small RMSE since there are few obstacles, and the signal could propagate without significant distortion. The results are shown in Fig. 3.10. Compared to Fig. 3.9, where the signal can propagate without significant obstacles in the vicinity of the transmitter in an angle of almost 180 degrees in azimuth, the scenario in Fig. 3.9 allows the signal to propagate in almost every azimuth directions.

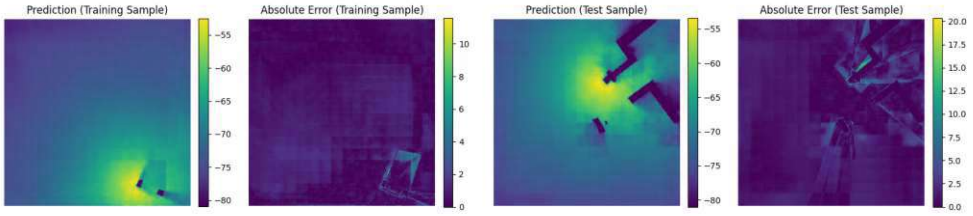


Fig. 3.10: Predictions of training and test sample that have the highest street share

The results are given as follows.

	Training Sample	Test Sample
RMSE	0.8876dB	1.7003dB
r_{street}	0.9983	0.9587

Although fewer objects than in Fig. 3.9 and the signal is more likely to perform free space propagation in almost every azimuth direction, the predictions have larger RMSE in both cases.

Predictions of training and test samples that show the highest RMSE are shown in Fig. 3.11.

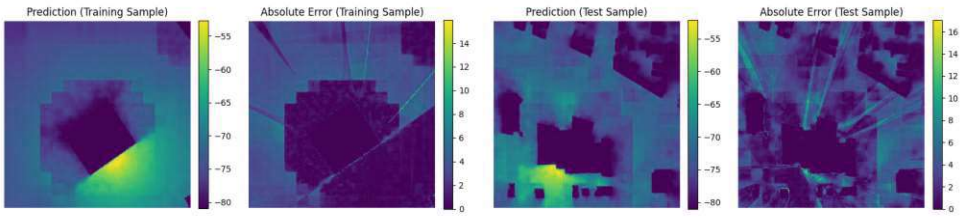


Fig. 3.11: Predictions of training and test sample that have the highest RMSE

The following table gives the RMSE and corresponding street shares.

	Training Sample	Test Sample
RMSE	2.9388dB	3.5911dB
r_{street}	0.9120	0.8

Although these samples have a higher street share than the samples given in Fig. 2.6 and Fig. 2.7, the network is not able to predict the signal strength with reasonable accuracy. Oddities in the structures of the shadows are prominent effects of the prediction. They appear to be displayed at low resolutions and have a squared structure, which is not observable in Fig. 3.7. In addition, the error map shows the deviation from the ground truth in some areas further away from the antenna. Unfortunately, the error map shows the absolute error. The information is lost if the prediction for a pixel is smaller or larger than the ground truth. For example, in figure 3.11, where the building should cast shadows, common sense suggests that the UNet predicts higher receive power in certain areas.

These samples with the highest RMSE demonstrate that the prediction accuracy depends on many factors, such as antenna placement, building placement, and street area on a given map. Figure 3.12 shows the RMSE and the street share for each prediction on each training and test sample and shows no clear correlation between a higher street share and the RMSE. The variation in RMSEs for similar street shares indicates that buildings or antenna placements play a critical role in prediction accuracy. For the training samples, the predictions with the largest RMSE are from samples with a relatively high street share (around $r_{\text{street}} = 0.8$ and $r_{\text{street}} = 0.9$). Furthermore, the RMSE shows more variation with higher street shares. RMSEs of predictions from test data samples show a slightly decreasing behavior for the higher street areas.

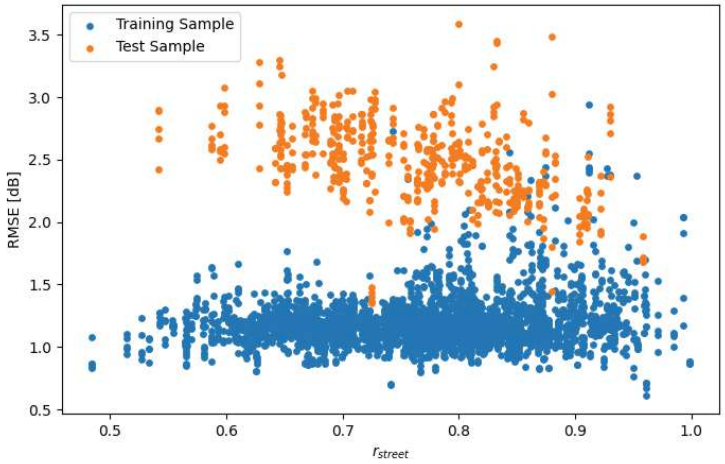


Fig. 3.12: This plot shows the RMSE and the corresponding street share r_{street} for each prediction of each training and test data sample.

Finally, R2 is

$$\boxed{\text{R2} = 0.8836,}$$

for training data samples, and it shows a small improvement to SegNET, where the R2 score is R2 = 0.8669, and for test samples, it is

$$\boxed{\text{R2} = 0.6461.}$$

R2 score for test samples is slightly better compared to R2 = 0.64, what the SegNET delivers.

3.3 Summary

- Two types of neural networks, namely SegNET and UNet, will be analyzed for signal strength prediction tasks
- SegNETs consist of an encoder and a decoder. These are a sequence of (transpose) convolutional layers. Instead of pooling or dropout layers, striding fulfills the task of increasing or decreasing the feature maps.
- UNets are similarly structured, but UNets have a skip connection between the corresponding encoder and decoder layers.
- UNets perform slightly better than SegNETs, even though both networks have the same number of layers and the configurations of each corresponding layer of both networks are the same.

- The accuracy of the prediction depends on the urban scenario. It is harder to obtain accurate predictions when the signal has many obstacles in its path that cause shadowing, scattering, reflection, and other phenomena.

Chapter 4

Measurement Channel for Local Calibration

In this analysis, the question arises of whether the UNet can improve the prediction when it receives additional information about the signal strength across the city map. Methods such as crowd sensing (CS) can be applied to investigate the wireless properties in a cell. These methods exploit the property that these devices are distributed throughout the urban area and can collect information about signal quality and location data and provide this local information to the radio network [36]. This information provides the UNet knowledge about the signal strength at specific locations and can be used to calibrate it implicitly.

This chapter will examine the performance of this approach by using different amounts of local information. A comparison of a UNet with a measurement channel versus one without highlights the increased accuracy gained from additional signal strength information. Adding local information to the network will support the network with information about the signal strengths in some locations and thus improve the accuracy of the prediction. Due to the accessibility of measurements across the city map, baseline generation is now feasible. This information on the signal strength serves as support points and observations. Baselines provide a foundation for evaluating the prediction accuracy of the UNet model.

4.1 Additional Measurement Input

A crucial assumption is the correctness of local signal strength information obtained from measurements, i.e., the signal strength values of specific positions on the map equal the ground truth at their positions. Modification of the network allows signal strength information to pass through. The most straightforward implementation is to increase the input by one channel, i.e., including a third feature map dimension and increasing the dimension of the input array of size $256 \times 256 \times 2$ by one $256 \times 256 \times 3$, which then requires an additional map of the same size as the other input maps. This map stores all the measurement information for a particular sample. A sketch is given in Fig. 4.1. This work investigates the effect of this additional channel on the learning curves and the prediction.

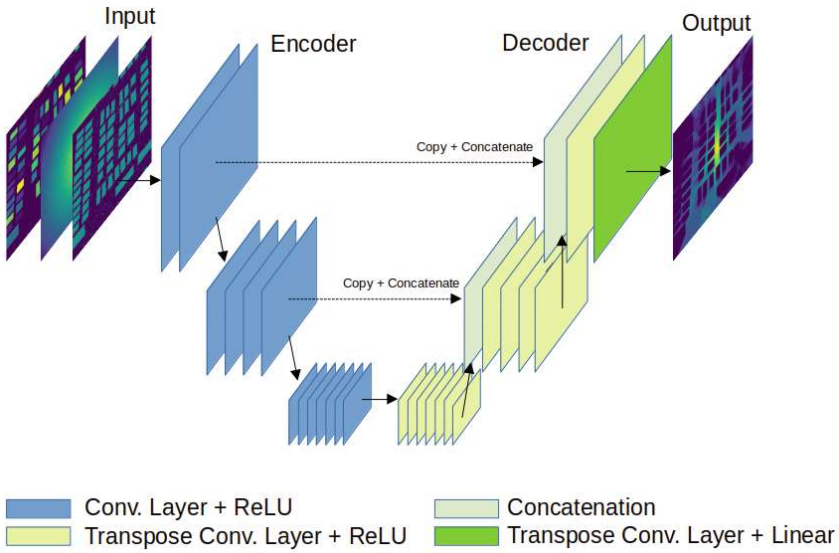


Fig. 4.1: Sketch of a UNet with three-channel input and single-channel output. The input consists of the two maps that have been used and the measurement map. The output is the signal strength prediction for a given urban scenario and antenna setting.

4.1.1 Measurement Map

An array, namely *measurement map*, stores the signal strength information and the locations of the measurements. The additional channel for the measurement map is called the measurement channel. Thus, the network has to process an additional image with 256×256 pixels. Therefore, the input size of the first UNet layer is $256 \times 256 \times 3$. Methods to generate this map are presented in [17], [37] and will be briefly summarized here. The process of creating a measurement map, denoted as $\boldsymbol{X}_{\text{measurement}}$, begins with a matrix $\boldsymbol{M}$ with elements

$$\boldsymbol{M}_{kl} = \begin{cases} 1, & \text{if position } [k,l] \text{ is measured} \\ 0, & \text{else} \end{cases} . \tag{4.1}$$

Assuming that a simulation exists for a specific urban scenario, declared as the ground truth $\boldsymbol{Y}$, the following equation

$$\boldsymbol{X}_{\text{measurement}} = \boldsymbol{M} \circ \boldsymbol{Y}, \tag{4.2}$$

simulates measurements, including their positions, where $\circ$ represents a Hadamard product, which is an element-wise product. This thesis utilizes the following definition for the measurement map, which is given by

$$\boldsymbol{X}_{kl,\text{measurement}} = \begin{cases} \boldsymbol{Y}_{kl}, & \text{if position } [k,l] \text{ is street area and measured} \\ 0, & \text{if position } [k,l] \text{ is building area} \\ -1, & \text{else} \end{cases} . \tag{4.3}$$

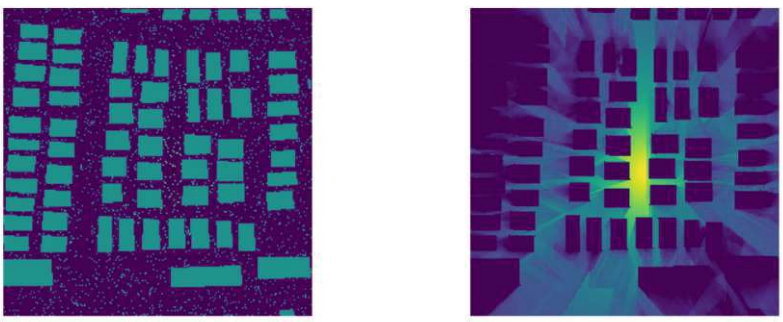
In contrast to (4.2), the measurement map additionally contains information about buildings' locations. Knowing the exact placement of buildings in an urban scenario justifies this choice. Since signal strength inside buildings is irrelevant in this thesis, the corresponding values are zero. Unmeasured street areas are set to -1 to inform the CNN that there is no information about that region. The authors in [17] have done an analysis with measurements of about 5% of the area and were able to generate complete radio maps with reasonable error. Therefore, this measurement amount is the reference in this simulation. To obtain measurements covering about 5% of the road area, first recall the set $\mathcal{N}_{j,\text{streets}}$, the set of index tuples $[k,l]$ of a sample j belonging to the road area, and $\mathcal{N}_{j,\text{building}}$, the set of index tuples $[k,l]$ of a sample j belonging to the building area. Then, for all elements of the measurement map $\boldsymbol{X}_{kl,\text{measurement}}$ that belong to the street area $[k,l] \in \mathcal{N}_{j,\text{streets}}$ are drawn from a probability mass function (pmf) $P(\boldsymbol{X}_{kl,\text{measurement}})$ with

$$\begin{aligned} P(\boldsymbol{X}_{kl,\text{measurement}} = \boldsymbol{Y}_{kl}) &= 0.05 \\ P(\boldsymbol{X}_{kl,\text{measurement}} = -1) &= 0.95, \end{aligned} \tag{4.4}$$

while the elements with indices $[k,l] \in \mathcal{N}_{j,\text{buildings}}$ are $\boldsymbol{X}_{kl,\text{measurement}} = 0$. A more general formulation for any probability that the street area position $[k,l]$ is measured is

$$\begin{aligned} P(\boldsymbol{X}_{kl,\text{measurement}} = \boldsymbol{Y}_{kl}) &= p \\ P(\boldsymbol{X}_{kl,\text{measurement}} = -1) &= 1-p \end{aligned} \tag{4.5}$$

Using these expressions ensures that the measurement covers, on average, 5% or p of the street regions. In this thesis, this value is termed as normalized measurement density (NMD), referring to a measurement density for a given street area. It gives the expected amount of measurements for a given sample. Also, when defined this way, the measurements are distributed over the entire street area of a sample. An example is given in Fig. 4.2a, while Fig. 4.2b shows the corresponding ground truth. The measurement map shows the signal strength values of a couple of points evenly distributed over the street area. It also shows silhouettes of buildings that have a signal strength of zero.



(a) Measurement Map **(b)** Ground Truth

Fig. 4.2: Measurement Map and Ground Truth

4.1.2 Training and Test Dataset

Training and test datasets generated in Sec. 2.3 are taken for this analysis. Since the generation of new sets in future simulations relies on these sets, this thesis labels the training dataset as $\mathcal{X}_{\text{Train}}$ and the test dataset as $\mathcal{X}_{\text{Test}}$. Each of the samples of the training dataset $\mathcal{X}_{\text{Train}}$ and test dataset $\mathcal{X}_{\text{Test}}$ contains three maps, namely the building information map, the antenna information map and the ground truth, where the building information map and the antenna information map form the input of a sample, i.e. $\boldsymbol{X} = (\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}})$, while the ground truth $\boldsymbol{Y}$ is used for training or evaluating the predictions. The construction of the training and test dataset with measurement map was simply the addition of a corresponding measurement map $\boldsymbol{X}_{\text{measurement}}$ to each sample from $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Test}}$. The Alg. A.1 describes the algorithm for its implementation. The input of a sample from the new set is now $\boldsymbol{X} = (\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}}, \boldsymbol{X}_{\text{measurement}})$. For this analysis, each sample has NMD of NMD = 5%, which will be called $\mathcal{X}_{\text{Train},5\%}$ and $\mathcal{X}_{\text{Test},5\%}$. Since the samples have different urban scenarios, the expected number of measurements depends on the street share. The first step to estimate the expected average number of measurements for a randomly selected sample is to determine the average street share for all samples, which is $\bar{r}_{\text{street}} = 0.767$. An urban scenario, discretized by $256 \times 256 = 65\,536$ pixels, has on average $0.767 \cdot 65\,536 = 50\,266$ pixels of street area. With an NMD of NMD = 5%, the expected amount of information about signal strength and its location is 2513. In this implementation, the training and test datasets have the same urban environment and antenna placement in the same order as their corresponding training and test datasets $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Test}}$. This choice simplifies the comparison of the three-channel UNet with the two-channel UNet of section 3.2.

4.1.3 Network Training

The network with an additional measurement channel was trained with samples from a training dataset $\mathcal{X}_{\text{Train},5\%}$. The training was made with epochs = 200, batch size = 16 and a learning rate of 10^{-3}. The learning curves are shown in Fig. 4.3 together with the learning rate of a two-channel UNet without measurement support as semitransparent curves. It shows that the training loss and the validation loss of the UNet with measurement support has a steeper decline at the beginning of the training process than the UNet without measurement support and a smaller RMSE at the end. In particular, the measurement-assisted network significantly reduces the loss of test samples. The behavior in terms of convergence, steepness, and noise of the learning curves of the measurement-supported UNet are similar to those of the non-measurement-supported UNet. The steepness learning curve for training samples decreases with an increasing number of epochs. However, this learning curve stays within the selected epoch range, while the learning curve for test samples converges at a very early stage of training.

The training loss after training is 0.6626dB for the three-channel input UNet and 1.1348dB for the two-channel input UNet, which is a reduction of 41.61%. The validation loss is reduced from 2.2256dB for a two-channel input UNet to 1.2166dB for a three-channel input UNet, which is a reduction of 45.34%.

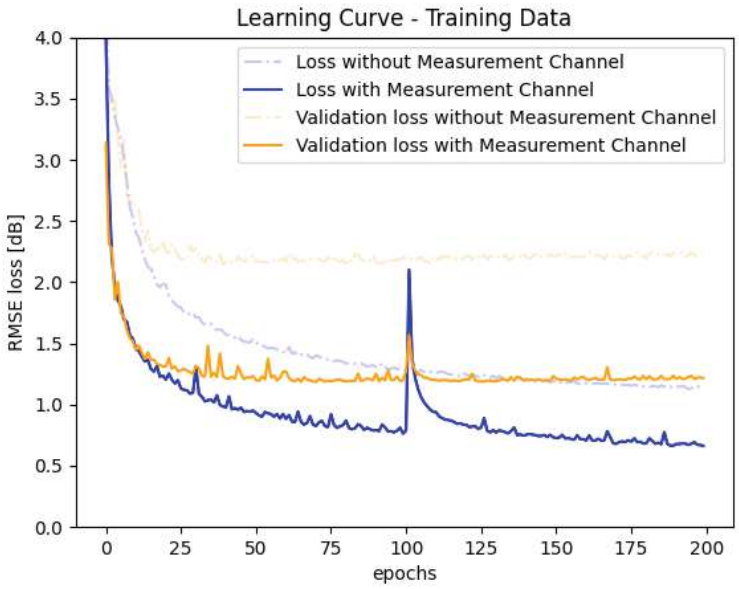
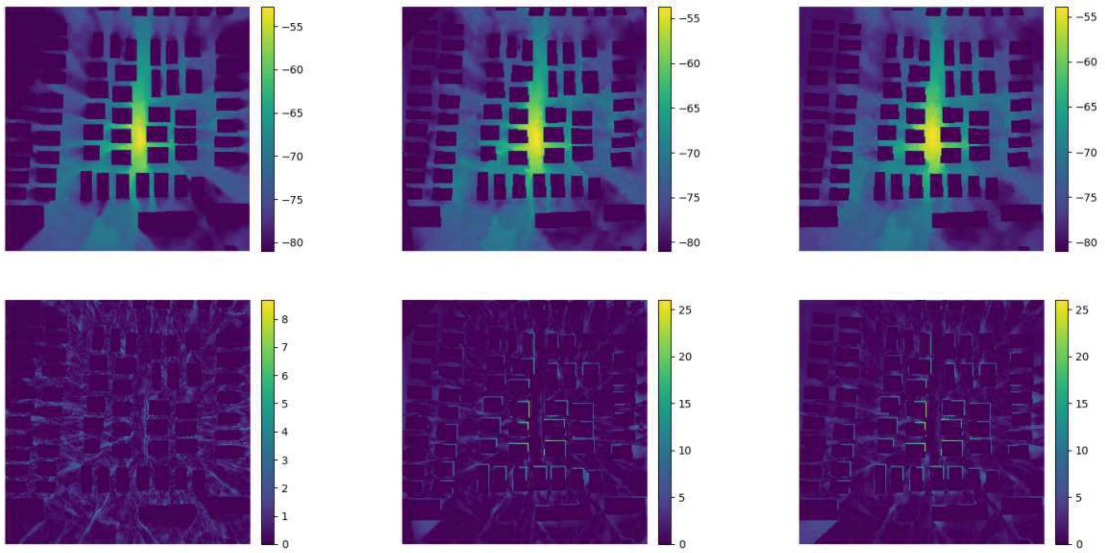


Fig. 4.3: Learning curves for a measurement assisted and not a non-assisted UNet

4.1.4 Prediction Results

First, the prediction of a training sample whose building and antenna information is shown in Fig. 2.6 will be presented. Additionally, this sample has a measurement map with NMD = 5%. This sample is known to the network due to the training. The prediction result is in the upper image in 4.4a, while the lower image shows the error map, i.e., the pixel-wise absolute error. Since exact information about the signal strength is available at some locations, linear interpolation, and KNN regression can be performed to generate baselines for benchmarking the UNet. The algorithms are in Alg. A.2 and Alg. A.3. The baselines and the corresponding error maps are in 4.4b and 4.4c. All methods show reasonable signal strength maps with detailed illumination and shadows. However, the error maps show that linear interpolation and KNN regression have problems calculating signal strength near buildings, possibly due to incorrect building detection. They also show deviations from the ground truth at the edge of shadows. The UNet error map does not show significant deviations from the ground truth compared to other methods[1].

[1]Note that all error maps have the same scale, so these maps are comparable to each other.



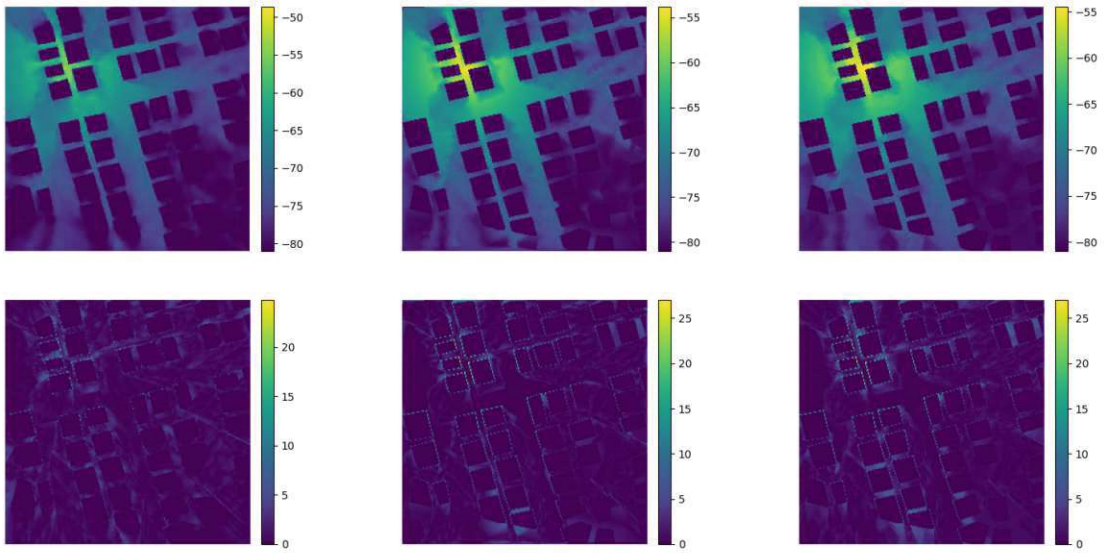
(a) UNet Prediction **(b)** Linear Interpolation **(c)** KNN Regression

Fig. 4.4: The first row shows the signal strength map that is generated with a UNet, linear interpolation, and KNN regression. The second row shows the error map of each method.

The RMSE of the UNet prediction prediction, linear interpolation, and KNN regression are given in the following table.

	UNet	Linear Interpolation	KNN Regression
RMSE	0.7837dB	2.3118dB	2.5110dB

Overall, UNet produces the most accurate predictions, with minor deviations and generally small RMSE. One can argue that the CNN has seen this sample during training and, therefore, can easily predict the signal strength for this sample, while other methods have to rely solely on measurements. Therefore, a test sample with a similar building-to-street ratio, shown in Fig. 2.7, will be chosen with a measurement map with NMD = 5% for the following prediction. The network has never seen this sample, i.e., it has to generate a prediction without knowing the ground truth. In contrast to the prediction of training data samples, it is more comparable to other methods used in this section. The results are in Fig. 4.5. In Fig. 4.5a, some light spots can be found, especially around buildings, that indicate deviation from the ground truth in these regions. The same applies for baselines.



(a) UNet Prediction **(b)** Linear Interpolation **(c)** KNN Regression

Fig. 4.5: The first row shows the signal strength map that is generated with a UNet, linear interpolation, and KNN regression. The second row shows the error map of each method.

The RMSE of each method is given in the following table

	UNet	Linear Interpolation	KNN Regression
RMSE	1.5468dB	1.9595dB	2.0994dB

As expected, the RMSE of the UNet prediction of this test sample is more significant than that of a training sample. Furthermore, it is smaller than the RMSE of the baselines, which interestingly is smaller than the RMSE of the baselines of the training sample discussed earlier. The linear interpolation and the KNN regression do not depend on whether the sample is from the training or the test dataset but only on the support points and the observation, respectively.

The overall performance of this three-channel input UNet, supported by measurements with NMD = 5%, will be expressed in terms of averaged RMSE separately for training and test samples. For 3000 training samples from the training dataset, the averaged RMSE of each prediction RMSE is

$$\boxed{\text{RMSE} = (0.7099 \pm 0.1)\text{dB}}$$

whereas for the 500 test samples from the training dataset, it is

$$\boxed{\text{RMSE} = (1.2693 \pm 0.2421)\text{dB}.}$$

4.2 UNet for Fixed Measurement Density

So far, the focus was on UNet for samples with measurement maps with a specific NMD, which in this case was NMD = 5%. In this section, the investigation will focus on variations in the NMD, with two approaches available. Analysis of the behavior of the UNet for different NMD enables the investigation of the effect of the variations in the NMD. The first approach is generating multiple training and test datasets, each with different NMD and where each measurement map of a training and the corresponding test dataset has the same particularNMD. The second approach will be part of the discussion in the next section.

The training of the UNet will take place with samples from a training dataset with a specific NMD, and it will predict samples with the same NMD. For example, a UNet involves the training dataset that contains samples with NMD = 1% for training purposes and predicts samples with NMD = 1%. Then, the performance of the UNet will be analyzed by evaluating the predictions. Next, the UNet will perform training with a training dataset with higher NMD and undergo performance evaluations.

The process will continue until the set covers the specified NMD range. From expectations, the more measurements passed through the network, the more knowledge about the signal strength the UNet receives to generate more accurate predictions.

4.2.1 Training and Test Dataset

These sets are the basis for analyzing the UNet for training and test data sets with different NMD. Based on the training $\mathcal{X}_{\text{Train}}$ and test datasets $\mathcal{X}_{\text{Test}}$ created in Section 2.3[2], training and a test dataset was created for each NMD $\in \mathcal{N}_{\text{NMD}} = \{1\%, 2\%, 5\%, 10\%, 20\%\}$, which is called $\mathcal{X}_{\text{Train,NMD}}$ and $\mathcal{X}_{\text{Test,NMD}}$, where NMD $\in \mathcal{N}_{\text{NMD}}$. For training and dataset with certain NMD, a measurement map $\boldsymbol{X}_{\text{measurement}}$ was created with this NMD for each sample from $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Test}}$. Then $\boldsymbol{X}_{\text{measurement}}$ was added to the input sample of $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Test}}$ to obtain $\boldsymbol{X} = (\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}}, \boldsymbol{X}_{\text{measurement}})$ for a training $\mathcal{X}_{\text{Train,NMD}}$ and test dataset $\mathcal{X}_{\text{Test,NMD}}$ with a certain NMD. Due to this choice of implementation, the sets $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Train,NMD}}$, with NMD $\in \mathcal{N}_{\text{NMD}}$, have the same set of urban scenarios in the same order. They only differ in their measurement maps depending on the NMD $\mathcal{X}_{\text{Train,NMD}}$. The same is true for the training datasets $\mathcal{X}_{\text{Test}}$ and $\mathcal{X}_{\text{Test,NMD}}$, with NMD $\in \mathcal{N}_{\text{NMD}}$. Examples of measurement maps with different NMD for a given urban scenario and antenna placement can be seen in Fig. 4.6.

These training and test datasets will be reused, i.e. $\mathcal{X}_{\text{Train}}$, $\mathcal{X}_{\text{Test}}$, $\mathcal{X}_{\text{Train,NMD}}$ and $\mathcal{X}_{\text{Test,NMD}}$, where NMD $\in \mathcal{N}_{\text{NMD}}$, mostly for prediction, in further analysis. Therefore, labels will be added to these sets to avoid confusion. In some cases, a UNet trained with specific training datasets predicts a sample with the same urban environment contained in the dataset that has been used

[2]These are sets that have input samples consisting of two maps $(\boldsymbol{X}_{\text{building}}, \boldsymbol{X}_{\text{antenna}})$ and their corresponding ground truth $\boldsymbol{Y}$.

for training but with a different measurement map. So, the sample is a training sample from a different training dataset. Therefore, this training dataset will be labeled as *known environment*, since the urban scenarios are known to the network, and the test dataset accordingly as *unknown environment*.

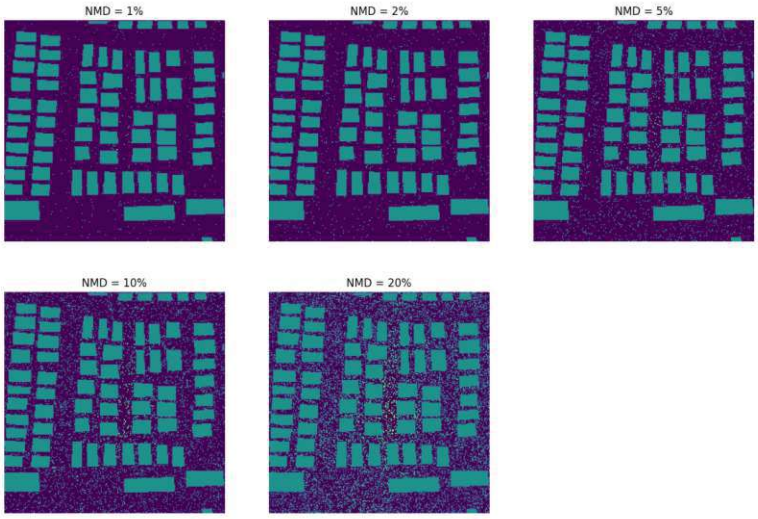
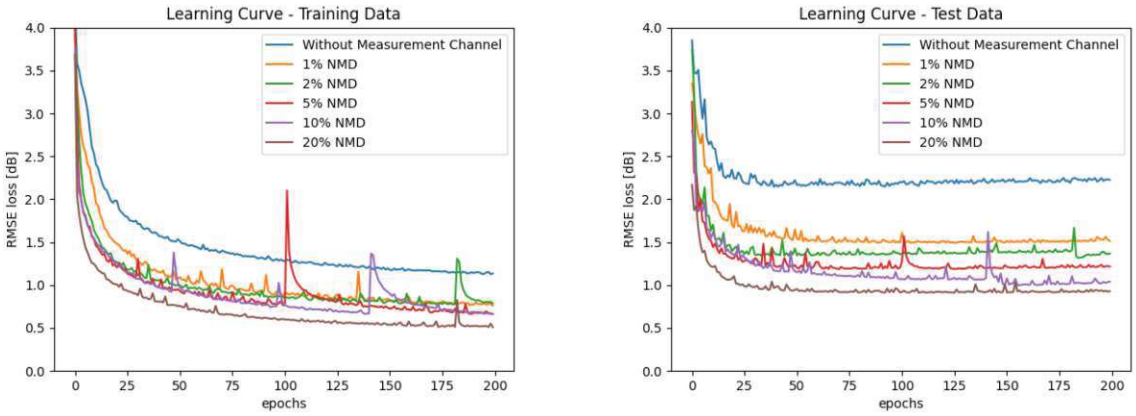


Fig. 4.6: Measurement maps for different NMD for an exemplary urban environment.

4.2.2 Network Training

Since there are five training and five test datasets with different NMD from the set $\mathcal{N}_{\text{NMD}}$, the training of the UNet was carried out with these different training datasets $\mathcal{X}_{\text{Train,NMD}}$, with NMD $\in \mathcal{N}_{\text{NMD}}$, and the learning curves for each dataset was examined. The results of training with 200 epochs and a batch size of 16 are in Fig. 4.7, which shows the learning curves for training data for different NMD in Fig. 4.7a and the learning curves for test data for different NMD in Fig. 4.7b. Furthermore, the Fig. 4.7 shows the learning curves for the two-channel UNet for comparison. In both plots, noisy behavior in the learning curves is present. In both plots, noisy behavior in the learning curves is present. In this training process, large spikes are common in the learning curves for the training dataset, which can lead to a higher loss for training data sets with larger NMD than for lower NMD. For example, training the UNet with training samples from $\mathcal{X}_{\text{Train},2\%}$ has a higher final loss than training with training samples from $\mathcal{X}_{\text{Train},1\%}$. Test data samples share the same behavior, but the spikes are not as pronounced as for training samples, and the final validation loss is lower for samples with larger NMD. The validation losses converge around 50 epochs, while the training losses do not converge in the epoch range.



(a) Learning Curves for Training Data **(b)** Learning Curves for Test Data

Fig. 4.7: Learning Curves for various NMD

4.2.3 Prediction Results

In Section 4.1.4, a UNet was analyzed for a selected sample from a training and a test dataset with NMD = 5%. This section completes the remaining NMD from $\mathcal{N}_{\text{NMD}}$. A UNet trained with samples from $\mathcal{X}_{\text{Train,NMD}}$ with a specific NMD performed the prediction of a training sample from $\mathcal{X}_{\text{Train,NMD}}$ with the same NMD. Furthermore, this sample has the same urban environment with the exact antenna placement as the training sample in Section 4.1.4, which is essentially Fig. 2.6. The following Tab. 4.1 shows the RMSE of the prediction and the baselines for NMD from $\mathcal{N}_{\text{NMD}}$ for this particular city map. Here, the RMSE does not decrease monotonically with increasing NMD.

Tab. 4.1: RMSE for a particular training sample with different NMD

NMD	UNet	Linear Interpolation	KNN Regression
1%	0.8682dB	2.6919dB	2.9705dB
2%	0.8562dB	2.4833dB	2.7499dB
5%	0.7837dB	2.3118dB	2.5110dB
10%	0.7723dB	2.1189dB	2.2792dB
20%	0.6272dB	2.0981dB	2.1854dB

The same analysis for the test sample, where the urban environment and antenna placement are in Fig. 2.7, delivers Tab. 4.2. This table demonstrates for unknown urban environment that the more information about the signal strength the UNet receives, the more accurate the predictions it can generate. Also, the baselines become more accurate as the linear interpolation and KNN regression have access to more information. For small NMD, i.e., when the measurement map contains sparse information about the signal strength, the UNet can still generate a more accurate prediction than the baseline generation methods, since it can rely on the model.

Tab. 4.2: RMSE for a particular training sample with different NMD

NMD	UNet	Linear Interpolation	KNN Regression
1%	2.0367dB	2.5811dB	2.4710dB
2%	1.8221dB	2.2602dB	2.2435dB
5%	1.5468dB	1.9595dB	2.0994dB
10%	1.2352dB	1.7946dB	1.9020dB
20%	1.0749dB	1.7196dB	1.8077dB

As done before, the averaged RMSE determines the overall performance of the UNet for training and test samples. The performance of linear interpolation and KNN regression is the benchmark for comparison. The performance of these two methods is independent of whether the measurement map from which the support points and the observation belong to the training or the test dataset. Therefore, the performance of these methods will not be distinguished for training or test samples. For computationally efficiency, the averaged RMSE computation involved only a subset of the training and test dataset. The subset for a specific NMD contains 500 randomly selected samples from the training dataset and 250 samples from the same NMD test dataset. The average RMSE and variance followed from the calculation for each RMSE for every baseline derived from measurement maps in this set. For the averaged RMSE of the UNet for a given NMD, the complete training $\mathcal{X}_{\text{Train,NMD}}$ and test dataset $\mathcal{X}_{\text{Test,NMD}}$ was used. The result is displayed in Fig. 4.8. The corresponding table with numerical results is in Tab. B.1 for the UNet and Tab. B.2 for the baselines. On average, the RMSE decreases as the density of measurements increases. This behavior holds for training and test data and UNet prediction, linear interpolation, and KNN regression.

The UNet predictions for training and test samples have the smallest RMSE over any NMD compared to the baselines in the given range. Fascinatingly, even with small measurement densities, the UNet can generate more accurate predictions than linear interpolation and KNN regression methods. Even for test samples, the UNet yields a smaller averaged RMSE than the baselines. The UNet has learned the propagation characteristics from the existing radio map and can apply that knowledge to unknown urban scenarios. Furthermore, the averaged RMSE of the UNet for the training dataset does not show a monotonically decreasing behavior due to the noisy behavior and spikes during the training process.

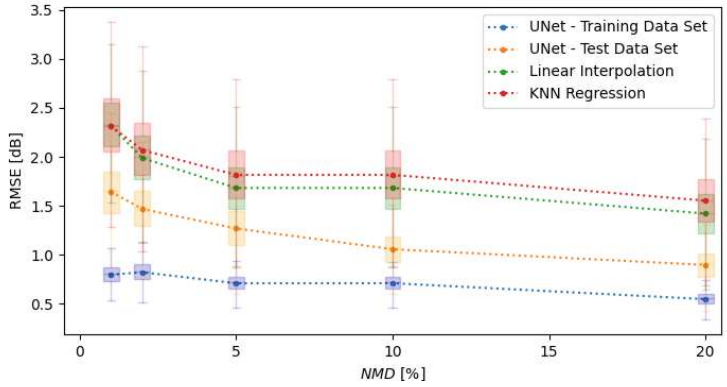


Fig. 4.8: Box plot of the RMSE over NMD for UNet, linear interpolation, and KNN regression. The training and test datasets are distinguished for UNet only. The points represent each method's averaged RMSE for a given NMD. Information about the distribution of the RMSE for a given NMD and methods can be retrieved from the box plot. The outliers are not displayed in this plot.

Next is a discussion on the R2 score of a UNet for training sample prediction, a UNet for test sample prediction, linear interpolation, and KNN regression. The overall R2 score for a given NMD is the average R2 score for each prediction or baseline. Again, a subset determines the computation of the R2 score due to computational efficiency. Results for the R2 score over the NMD can be seen in Fig. 4.9, where the UNet prediction of the training dataset has the highest R2 score over all NMD, which by definition means that the input explains most of the variation. The R2 score for the test dataset starts from a lower R2 score and increases with increasing NMD, but it is always smaller than that of the training dataset. Other methods, such as linear interpolation and KNN regression, start with a much lower R2 score. They increase with more measurements, especially at the beginning of the graph, but will never outperform the neural network.

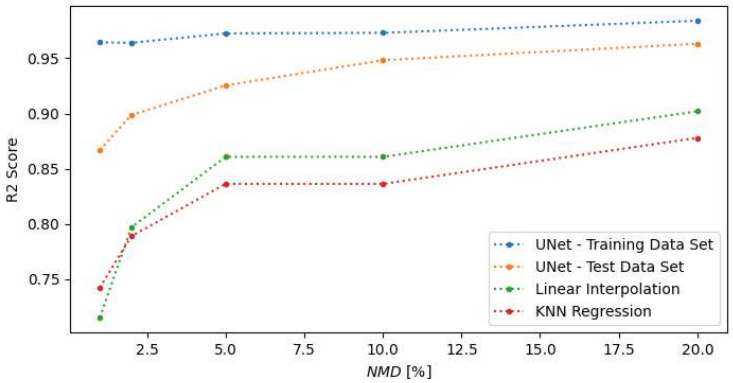


Fig. 4.9: Averaged R2 score over NMD for different methods

4.3 UNet for Variable Measurement Density

So far, this thesis has analyzed a UNet trained with samples, each with the same measurement density. Therefore, the network has learned to process measurement maps with a specific NMD. Usually, this network predicts samples from the same training or the test dataset with the same NMD. Some experiments have shown that if a sample has a different NMD than the training dataset used to train the network, then the network will generate considerably less accurate predictions, even if the measurement density is higher than that NMD. Therefore, this network will be called *NMD-specialized UNet* since it prefers measurement maps with NMD, which is exactly the NMD of $\mathcal{X}_{\text{Train,NMD}}$ used to train the UNet. The recommendation specifies that the test samples have the same NMD as the training samples. Therefore, the measurements for a region should fulfill this criterion in the real world. However, a more realistic scenario is that the density of measurements may vary from sample to sample, i.e., some samples contain more or less measurements per square meter in street areas than other samples. In this section, investigations focus on the performance of a UNet for samples with different NMD. This UNet is called *NMD-flexible UNet*.

4.3.1 Training and Test Dataset

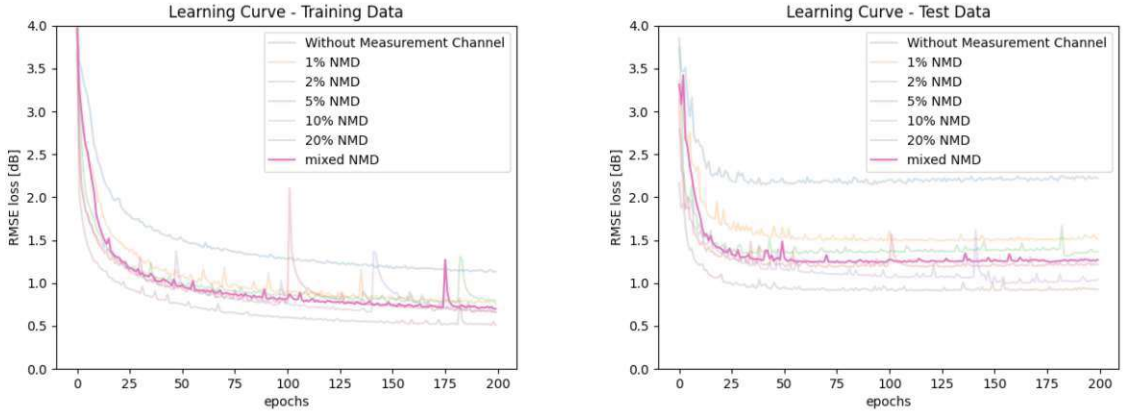
The training and test datasets discussed in this section served only the training and validation of the CNN. The training $\mathcal{X}_{\text{Train}}$ and test dataset $\mathcal{X}_{\text{Test}}$ created in Sec. 2.3 are the starting point. Again, a measurement map was added to each sample from these sets to obtain new training and test datasets, which is called $\mathcal{X}_{\text{Train,var}}$, $\mathcal{X}_{\text{Test,var}}$, to indicate the variation of the NMD of each sample in the sets. The urban environments of the samples in $\mathcal{X}_{\text{Train}}$ and $\mathcal{X}_{\text{Train, var}}$ are the same and have the same order. The same is true for $\mathcal{X}_{\text{Test}}$ and $\mathcal{X}_{\text{Test,var}}$. The generation of measurement maps took place with different NMD, and a uniform distribution determines the value for the NMD for each map. The uniform distribution has limits from 2% to 7%, i.e., $\text{NMD} \sim \mathcal{U}(0.02, 0.07)$. The network will learn how to process measurement maps with NMDs that are within this range. The choice of the range was, in a way, such that it is possible to reuse some samples that have smaller NMD than the lower limit, i.e. $\text{NMD} < 2\%$, and larger NMD than the upper limit, i.e. $\text{NMD} > 7\%$, to investigate how the UNet performs when the NMD of the sample is outside the range trained NMD range.

4.3.2 Network Training

The learning curve of the NMD-flexible UNet for the training dataset $\mathcal{X}_{\text{Train,var}}$ is shown in Fig. 4.10a and the validation using $\mathcal{X}_{\text{Trest,var}}$ is shown in Fig. 4.10b in pink color. Furthermore, the same learning curves from Sec. 3.2.1 for the UNet are shown for comparison and are named *Without Measurement Channel* in the legends. These curves represent the training loss and validation loss over epochs of a two-channel input UNet trained with $\mathcal{X}_{\text{Train}}$ and validated with $\mathcal{X}_{\text{Test}}$, which are sets of samples without measurement maps. In addition, this Fig. 4.10 shows the

learning curves for each NMD $\in \mathcal{N}_{\text{NMD}}$ of the NMD-specialized UNet trained with $\mathcal{X}_{\text{Train,NMD}}$ and validated with $\mathcal{X}_{\text{Test,NMD}}$. These are labeled with 1% NMD, 2% NMD, and so on.

The Fig. 4.10 shows that the training loss and the validation have similar behavior and have slightly higher losses than the losses from the NMD-specialized UNet that has trained with samples with NMD = 5%. Unfortunately, noisy behavior is present during the training phase.



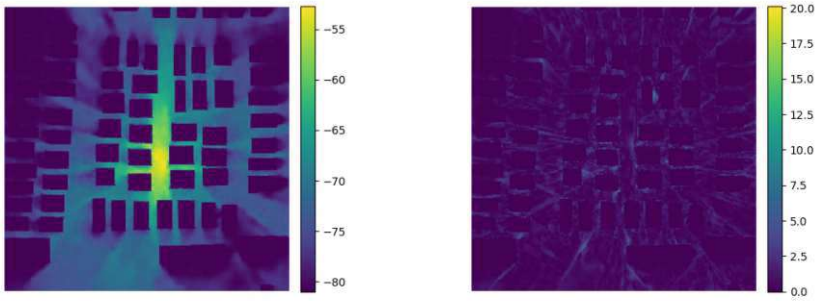
(a) Learning Curves for Training Data.

(b) Learning Curves for Test Data.

Fig. 4.10: Learning curve (highlighted) for training and test datasets with measurement cards with different NMD from NMD = 2% to NMD = 7%. The validation loss has also been computed with samples with different NMD. The legend calls this curve is called *mixed* NMD. The learning curves from the last section are also shown in semitransparent colors for comparison. These are learning curves from a two-channel input UNet and learning curves from NMD-specialized UNets, where each of which is trained with samples having the same NMD from the set $\mathcal{N}_{\text{NMD}}$.

4.3.3 Prediction Results

This NMD-specialized UNet explicitly predicts samples with measurement maps with NMD = 5% with high accuracy. The question arises about how the NMD-flexible UNet performs when it generates a signal strength prediction with samples with NMD = 5%. For a fair comparison, the training and test samples should have the same urban scenarios and antenna placements as the training and test samples in Sec. 4.2.3, which are depicted in Fig. 2.6 and Fig. 2.7. Since it is unlikely that $\mathcal{X}_{\text{Train,var}}$ contains this particular training sample with the same urban environment and antenna placement and an NMD of NMD = 5%, the corresponding training sample is from $\mathcal{X}_{\text{Train,5\%}}$ and it will be called *known environment with NMD* = 5%, because the NMD-flexible UNet knows the environment due to the training process, but the measurement map is unknown. The prediction result is given in Fig. 4.11a and the pixel-wise absolute error in Fig. 4.11b.



(a) Prediction **(b)** Absolute Error

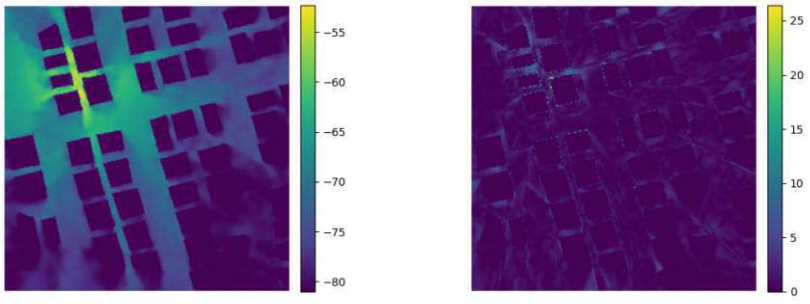
Fig. 4.11: Prediction of a *known environment* with measurement map with NMD = 5%.

Computing the RMSE allows a direct comparison of this prediction and the prediction of an NMD-specialized UNet with NMD = 5%. For this *known environment*, the RMSE is

$$\boxed{\text{RMSE} = 1.0747\text{dB},}$$

which is 0.291dB greater than the prediction of the NMD-specialized UNet with NMD = 5%. This NMD-specialized UNet has an advantage over NMD-flexible UNet because this *known environment* with NMD = 5% is contained in the training dataset $\mathcal{X}_{\text{Train},5\%}$, that was part of the training data set for the NMD-specialized UNet, therefore it has already seen this sample. Consequently, it can predict the signal strength with more accuracy.

A test sample from Fig. 2.7 with a measurement map with NMD = 5% serves for the following analysis. This sample is unknown to both networks, i.e., the NMD-specialized UNet and the NMD-flexible UNet. The NMD-flexible UNet produces a signal strength prediction that is shown in Fig. 4.12a and the pixel-wise absolute error is shown in Fig. 4.12b.



(a) Prediction **(b)** Absolute Error

Fig. 4.12: Prediction of an *unknown environment* with measurement map with NMD = 5%.

A measure to quantify the quality of the prediction is again the RMSE, and it is

RMSE 1.554dB,

moreover, it is only 0.0072dB larger than the prediction with the NMD-specialized UNet for NMD = 5%. Thus, both networks have similar accuracy in predicting signal strength for a *unknown environment*.

Next, the discussion focuses on the overall performance of the NMD-flexible UNet for each NMD $\in \mathcal{N}_{\text{NMD}}$. The averaged RMSE expresses this performance. This analysis distinguishes the averaged RMSE for *known environment* and *unknown environment*. Therefore, for each NMD $\in \mathcal{N}_{\text{NMD}}$ the averaged RMSE was determined for *known environments* by computing the RMSE of all predictions of all samples in the training datasets $\mathcal{X}_{\text{Train,NMD}}$. The process to compute the RMSE for *unknown environments* was similar and included $\mathcal{X}_{\text{Test,NMD}}$ for an NMD $\in \mathcal{N}_{\text{NMD}}$. Fig. 4.13 compares these results with the averaged RMSE of the NMD-specialized UNet for (un)known environments[3] for various NMD and the baselines. For clarity, these plots exclude the statistical characteristics. In addition, Fig. 4.13 compares the networks and the baseline. It shows that the NMD-flexible UNet generates predictions with lower expected accuracy for a known environment than when it is unknown. For larger NMD, the expected accuracy of the prediction seems to decrease when the NMD is well outside the trained range of $2\% < NMD < 7\%$. Speculatively, the prediction might have a larger RMSE for much higher NMD $> 20\%$ than the baseline methods using this NMD. In conclusion, the network cannot handle higher measurement densities than the NMD of the samples used for training. One can imagine how an NMD-specialized network would behave when it predicts samples with a different NMD than the NMD used for network training. The NMD-specialized UNet generates more accurate signal strength predictions for unknown environments than the NMD-flexible UNet, and the accuracy increases with larger NMD. However, the drawback of the NMD-specialized UNet is that it cannot handle an extensive range of NMD and must be trained for each NMD separately.

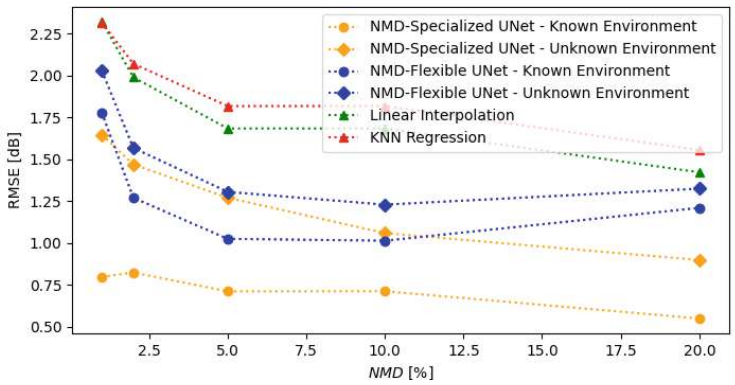


Fig. 4.13: This graph shows the averaged RMSE of predictions from NMD-specialized and NMD-flexible UNet of known and unknown environments for NMD $\in \mathcal{N}_{\text{NMD}}$. It also shows the RMSE of the baselines as a function of the NMD.

[3]Note that for the NMD-specialized UNet for a given NMD, the known environment with NMD is the training data set used to train that network, i.e. it has already seen those samples.

Finally, this paragraph compares the R2 scores with other methods for the NMD-flexible UNet, as shown in Fig. 4.14. The R2 score is highest when the samples have an NMD around the trained range of the NMD-flexible UNet, which is $2\% < NMD < 7\%$. Furthermore, when the environment is known, and NMD $\leq 10\%$, the R2 score of the prediction is higher than the R2 score of the prediction of the unknown environment. However, for NMD $= 20\%$, the model becomes worse for the known environment than for the unknown environment. From the behavior of the R2 score, it is possible that for NMD > 20, the R2 score will continue to decrease and become worse than the baselines.

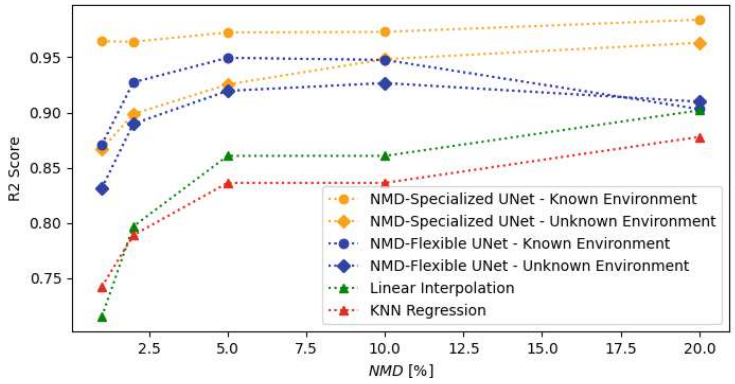


Fig. 4.14: R2 score for NMD-specialized and NMD-flexible UNet and baselines

4.4 Summary

- The prediction of a UNet becomes more accurate when supported with additional information about the signal strength.
- The information is encoded in the *measurement map*. The UNet will be trained with it to learn to process this additional information.
- The more information passed to the UNet, the higher the prediction accuracy. However, the accuracy increases slowly with NMD.
- The UNet prediction for a given NMD is more accurate than the baselines generated with linear interpolation and KNN regression using the information of the measurement map with this certain NMD.
- A UNet must learn to deal with samples with different NMD to handle it.

Chapter 5

Realistic Measurement Conditions

The uniform distribution of the measurements across the entire urban area is unrealistic, even for CS. People are not evenly distributed on the streets and often gather in busy areas. Section 5.3 presents a random walker that explores the streets and scans only parts of the urban environment, providing dense signal strength information for certain regions while leaving others unscanned. The performance of the UNet prediction will be investigated and compared to the baselines, where the information gathered by the random walker provides the support points and observations.

So far, the signal strength measurements are error-free. Realistically, however, any measurement is subject to error. A radio measurement device contains many components, such as a receiver antenna, amplifier, mixer for frequency conversion, filters, and many more, and each adds noise to the measurement. In addition, the information from the locations of the measurements can be erroneous due to GPS error. In the first two sections, i.e., Sec. 5.1 and 5.2, the measurements include distortion in signal strength and position, and these sections conduct the analysis of how the UNet performs prediction under this condition. The predictions with baselines generated with methods using distorted measurements serve as a benchmark for the network.

5.1 Signal Power Noise

In this section, the signal power contains white noise, and investigations on the UNet allow an analysis of how it handles the distortion of measurements. Therefore, a method will generate a training and test data set, where each sample contains a measurement map with distorted signal power measurements. An additive white Gaussian noise given by the variance σ^2 determines the intensity of this distortion. A network trained with a dataset where each measurement suffered from the same noise with variance σ^2 will be called a *signal noise specialized UNet*. This UNet will predict samples with measurement maps where the measurement is disturbed by white noise with the same variance σ^2. Only the behavior of the signal power noise is interesting for this analysis. Therefore, the position measurement is considered correct.

5.1.1 Training and Test Dataset

The basis was the training $\mathcal{X}_{\text{Train},5\%}$ and test dataset $\mathcal{X}_{\text{Test},5\%}$ from Section 4.1.2, which already contain measurement maps with an NMD of MND = 5%. The measurement map delivers

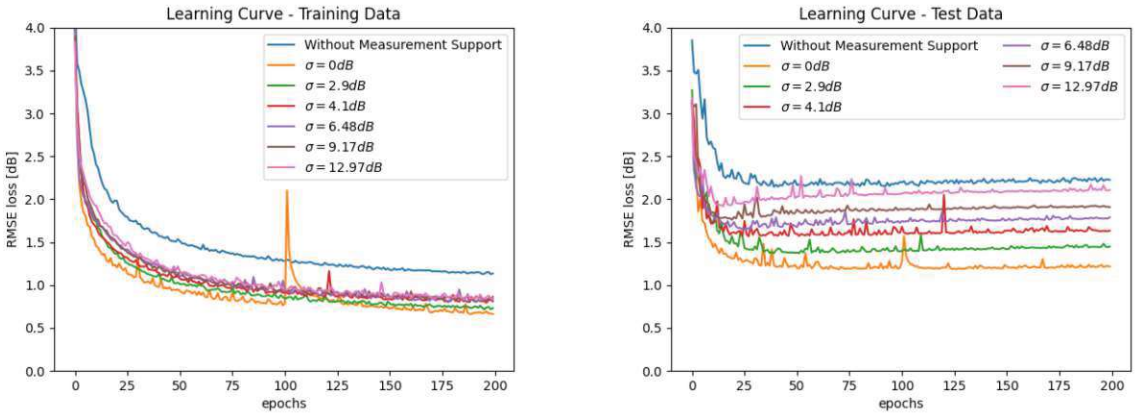
information about the measurement, i.e., the position of the measurement on the map and its normalized signal strength value. A method extracted this information and added a zero mean random value $\mathcal{N}(0, \sigma^2)$. Any values below zero due to addition with a negative random value, and therefore below the noise floor, were set to -1. The interpretation is that they are outliers, and the evaluation usually excludes them. Therefore, the more variance in the noise, the more information is lost, and the NMD will be slightly lower than the original. Training and test datasets were created for each $\sigma^2 \in \mathcal{N}_\sigma^2 = \{0.01, 0.02, 0.05, 0.1, 0.2\}$. The resulting training dataset will be denoted as $\mathcal{X}_{\text{Train,S},\sigma^2}$ and the test dataset as $\mathcal{X}_{\text{Train,S},\sigma^2}$, where S in the index indicates the distorted signal power and $\sigma^2 \in \mathcal{N}_\sigma^2$. Using this implementation, the training data sets $\mathcal{X}_{\text{Train,S},\sigma^2}$, $\sigma^2 \in \mathcal{N}_\sigma^2$ and the training dataset $\mathcal{X}_{\text{Train},5\%}$ have the same urban scenarios in the same order. Furthermore, each corresponding measurement map of $\mathcal{X}_{\text{Train,S},\sigma^2}$, $\sigma^2 \in \mathcal{N}_\sigma^2$, and $\mathcal{X}_{\text{Train},5\%}$ has identical measurement distributions for a given urban scenario and antenna placement. The same is true for the test datasets. In other words, for a given building and antenna information map, the locations of the measurements are the same in each training and test dataset. The measurements of the same sample of each training or test dataset and the same position on the map differ only in their perturbations, i.e., the measured signal strengths might have different values.

It is possible to create datasets by generating new measurement maps from scratch. This process would result in datasets with a different number and distribution of measurements for a given urban scenario and antenna placement, which may also affect the prediction. Therefore, white noise was added to each existing measurement to study only the effect of measurement error on the prediction.

5.1.2 Network Training

This Section focuses on network training with the support of erroneous measurements. Note that there were no modifications to the ground truth. Since there are five training and test datasets, i.e., one for each $\sigma^2 = \{0.01, 0.02, 0.05, 0.1, 0.2\}$, UNet was trained for each of these σ^2. This UNet will be called *signal noise specialized UNet* since it predicts samples with measurement distortion with a given σ^2. The network trained with $epochs = 200$ and $batch\ size = 16$. The learning curves for training and test data samples are shown in Fig. 5.1. The left plot in this figure shows the learning curves of the training samples for different σ^2, and the right plot shows the learning curves of the test samples for different σ^2. The standard deviation in the dB scale indicates the noise intensity. Furthermore, the learning curves for training and test samples that are not distorted ($\sigma = 0$dB) and learning curves for a non-measurement supported UNet (labeled as *Without Measurement Support*) are shown for comparison. mi Due to the noisy behavior of the training sample learning curves, it is difficult to determine the losses of each UNet individually. However, these learning curves are all lower than the learning curves of non-measurement-supported training and more significant than the case where the measurements are error-free. The effect of measurement uncertainty is more apparent for the learning curves of the test data sample than for training data. The corresponding plot shows that for error-free

measurements, the (R)MSE loss is lowest over an extensive range of epochs, as expected, and the loss increases with increasing noise intensity σ. Fascinatingly, even the learning curve for test samples with large measurement perturbations, i.e., $\sigma^2 = 0.2$, shows smaller losses over an extensive range of epochs than the learning curve of a non-measurement-assisted UNet. A $\sigma^2 = 0.2$ variance corresponds to a standard deviation of $\sigma = 0.4472$, which is $\sigma = 12.97dB$ on a dB scale. Adding a random value with this standard deviation to a measurement would mean a multiplication by a factor of 19.815 on a linear scale. This validation loss for this noise intensity is smaller than that of UNet without a measurement channel, even if the additional measurement passed through the network contains significant uncertainties.

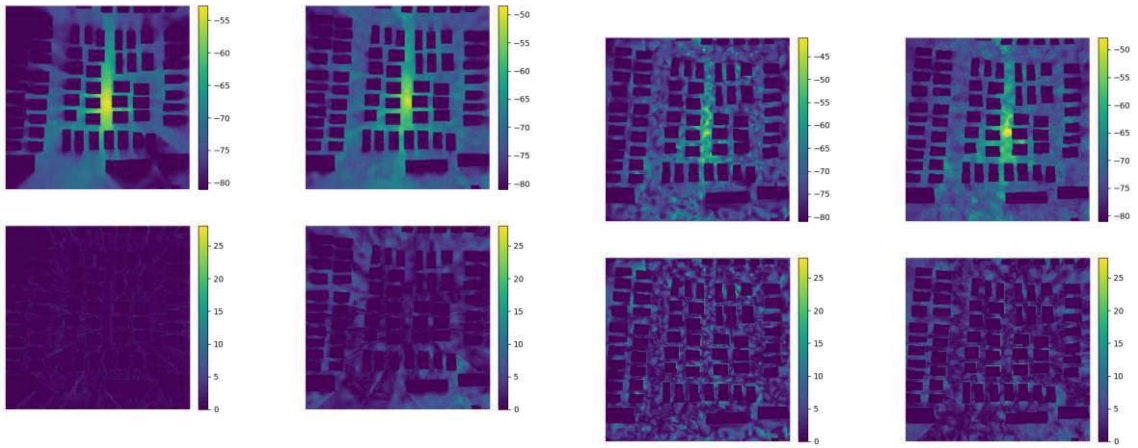


(a) Learning Curves for Training Data **(b)** Learning Curves for Test Data

Fig. 5.1: Learning Curves for different σ^2

5.1.3 Prediction Results

This paragraph presents the results of the UNet prediction of a sample of a known urban scenario, shown in Fig. 2.6. This sample includes a measurement map that is affected by an additional white noise with variance $\sigma^2 = 0.05$, i.e., the sample is in $\mathcal{X}_{Train,S}, 0.05$. First, a signal strength prediction was generated for this sample using a signal noise specialized UNet for $\sigma^2 = 0.05$. Then, a NMD-specialized UNet generated a prediction using this sample for comparison. The comparison also includes Baseline-generating methods such as linear interpolation and KNN regression. Predictions, baselines, and the corresponding error maps showing the pixel-wise deviation from the ground truth are shown in Fig. 5.2. Figure 5.3a shows the prediction of a noise-specialized UNet for this particular sample. The error map below shows that this particular UNet can compensate for the measurement uncertainties and produce a signal strength prediction with a slight deviation from the ground truth. A NMD specialized UNet is not trained on erroneous measurements, resulting in a noticeable deviation from the ground truth prediction, as shown in Fig. 5.2b. Figures 5.2c and 5.2d show that the uncertainties strongly affect the baselines, and it is not possible to generate proper baselines with either linear interpolation or KNN regression.



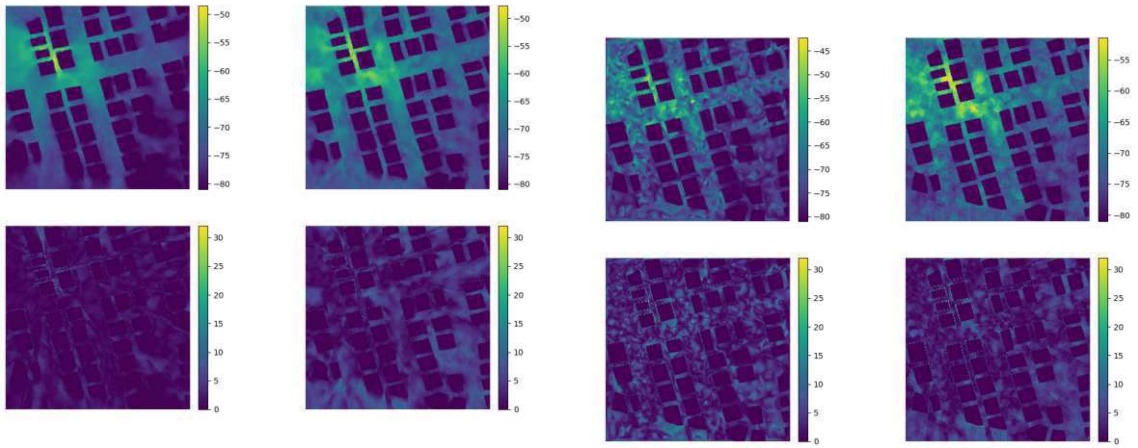
(a) Signal Noise Specialized UNet **(b)** NMD-Specialized UNet **(c)** Linear Interpolation **(d)** KNN Regression

Fig. 5.2: The first row shows the signal strength map for a known environment, that is generated with a Signal Noise Specialized UNet for $\sigma^2 = 0.05$, NMD-specialized UNet for MND = 5%, linear interpolation and KNN regression. The second row shows the error map of each method.

The RMSE for each method is given in the following table.

	Signal Noise Specialized UNet	NMD-Specialized UNet	Linear Interpolation	KNN Regression
RMSE	0.8868dB	3.6943dB	5.1630dB	4.4465dB

The predictions of an unknown environment sample shown in Fig. 2.7, will be analyzed, which additionally has a distorted measurement map with $\sigma^2 = 0.05$ that is in $\mathcal{X}_{Train,S}, 0.05$. Similar to the analysis of the prediction of the training sample, the question arises as to how a signal noise specialized UNet and an NMD-specialized UNet will perform. Furthermore, comparison includes linear interpolation and KNN regression. The visual result is in Fig. 4.5. As can be seen, the signal noise specialized UNet performs the best since it knows how to counteract measurement noise. This noise seems to confuse the NMD specialized UNet so that the signal strength prediction shows a blurry illumination of the signal strength, and buildings do not cast shadows with a clear, detailed structure.



(a) Signal Noise Specialized UNet **(b)** NMD-Specialized UNet **(c)** Linear Interpolation **(d)** KNN Regression

Fig. 5.3: The first row shows the signal strength map for an unknown environment, that is generated with a signal noise specialized UNet for $\sigma^2 = 0.05$, NMD-specialized UNet for MND $= 5\%$, linear interpolation and KNN regression. The second row shows the error map of each method.

The RMSEs given in the following table show the expected results. The signal noise specialized UNet has the highest accuracy for this sample. Interestingly, the performance of the NMD-specialized UNet is similar to the linear interpolation and is worse than the KNN regression.

	Signal Noise Specialized UNet	NMD-Specialized UNet	Linear Interpolation	KNN Regression
RMSE	2.0345dB	4.3537dB	4.3608dB	3.7708dB

After displaying the performance of the prediction and baseline generation for a single sample from the training $\mathcal{X}_{\text{Train,S,0.05}}$ and test dataset $\mathcal{X}_{\text{Test,S,0.05}}$, it is interesting to know the overall performance of these methods, which the averaged RMSE expresses. Each method exhibits different averaged RMSE over noise variance σ^2. The methods are signal noise specialized UNet, where this UNet was trained separately for each $\sigma^2 \in \mathcal{N}_{\sigma^2}$ and performed predictions for samples containing measurements with signal noise given by σ^2 of the training dataset used for training, a NMD-specialized UNet for MND $= 5\%$, linear interpolation and KNN regression.

The plot of the averaged RMSEs of the predictions for all known environments in $\mathcal{X}_{\text{Train,S},\sigma^2}$ and unknown environments in $\mathcal{X}_{\text{Test,S},\sigma^2}$ for each method and for given variances $\sigma^2 \in \mathcal{N}_{\sigma^2}$ is in Fig. 5.4. Findings show that the RMSE increases when the uncertainties of the measurement become more dominant, whereby linear interpolation and KNN regression are most susceptible to measurement uncertainty. At the same time, the signal noise specialized UNet is more resilient to this distortion because it has gained the ability to handle signal strength measurement

uncertainties through training. Furthermore, the noise confuses the NMD-specialized UNet. The RMSE curves of the predictions of a signal noise specialized UNet for known and unknown environments increase slowly with increasing σ^2. The fact that the measurements become less reliable explains this phenomenon. The network may reduce the influence of the measurement channel on the prediction and become more of a non-measurement-supported network, or it may transfer the distortion to the predictions. More research needs to be done in this area to provide more clarity. Numerical results are given tab B.4, B.5 and B.6.

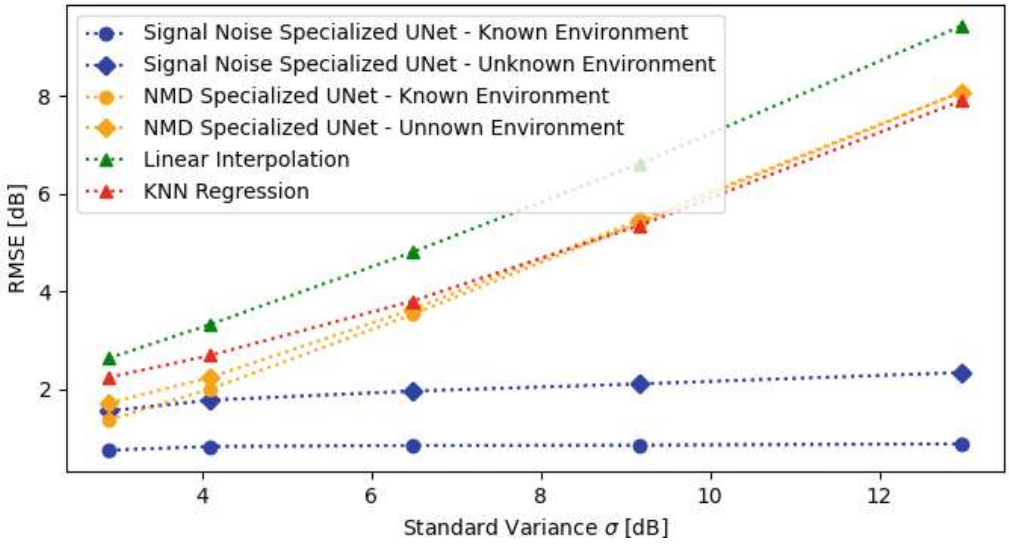


Fig. 5.4: Average RMSE over signal noise variance σ^2 for known and unknown environments and different methods.

The R2 score for each method is given in Fig. 5.5. It shows a high R2 score of signal noise specialized UNet predictions, and the R2 has a slight decay. At the same time, other methods, such as NMD-specialized UNET, linear interpolation, and KNN regression, start with smaller R2 scores and worsen with increasing measurement distortion, whereby the linear interpolation exhibits an inferior model than KNN regression and the NMD-specialized UNET shows the worst results for known environments. These methods achieve negative values when the distortion reaches a certain noise level of $\sigma > 5dB$.

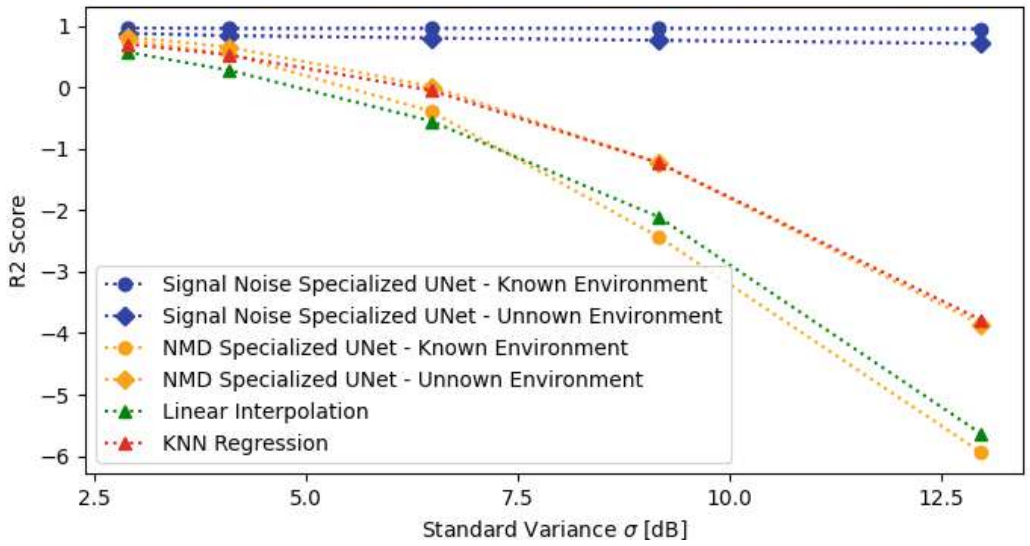


Fig. 5.5: Averaged R2 score over measurement uncertainty for different methods

5.2 Position Measurement Uncertainty

In real-world scenarios, positional measurement uncertainties may occur, providing a signal strength measurement slightly off the position where it is measured. Therefore, the examination will show the behavior of the UNet when the position of the signal strength measurement is perturbed while assuming that the signal strength measurement is accurate. A network will learn to compensate for these distortions, as in the previous section.

5.2.1 Training and Test Dataset

Based on the data sets $\mathcal{X}_{\text{Train},NMD}$ and $\mathcal{X}_{\text{Test},NMD}$ created in Sec, 4.1.2, which has MND = 5%, appropriate modifications were made to the measurement map of each sample to simulate position measurement error. First, a method identified all the elements corresponding to a signal strength measurement on the measurement map. The indices of the elements correspond to the coordinates of the measurement on the map. An array of length $N_{\text{measurement}}$ stored all coordinates, where $N_{\text{measurement}}$ is the number of measurements of a particular sample. Next, the method introduced positional perturbation by alternating the coordinates of the measurements to move them to a slightly different position. Therefore a two-dimensional multivariate Gaussian random vector $\tilde{\mathbf{n}} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2_{\text{pos}})$ was introduced, where

$$\begin{aligned} \boldsymbol{\mu} &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \\ \boldsymbol{\sigma}^2_{\text{pos}} &= \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}. \end{aligned} \tag{5.1}$$

The elements of the random vector $\tilde{\mathbf{n}}$ have zero means with variance σ_x^2 or σ_x^2 and are independent of each other. This analysis assumed that the perturbation in the x-direction is the same as in the y-direction, i.e., $\sigma_{\text{pos}}^2 = \sigma_x^2 = \sigma_y^2$ for simplicity. A non-zero mean $\boldsymbol{\mu} \neq \mathbf{0}$ could also be chosen as well as a non-diagonal covariance matrix $\boldsymbol{\sigma}_{\text{pos}}^2$ with σ_x^2 to simulate a drift in a specific direction, for example. The position coordinates for the measurement i are both integers and bounded between $(x_i, y_i) \in [0, 255] \times [0, 255]$, so it is necessary to convert $\tilde{\mathbf{n}}$, which is usually $\mathbb{R}^2$, to $\mathbb{Z}^2$. The simplest method is to round each element to the nearest integer.

For each measurement $i = 1, \dots N_{\text{measurement}}$ of a particular sample, a position noise vector $\mathbf{n}_i$ was generated and added to the coordinates of the measurement, i.e. $(\hat{x}_i, \hat{y}_i) = (x_i + n_{x,i}, y_i + n_{y,i})$. Due to the limited range of the coordinates $(\hat{x}_i, \hat{y}_i) \in [0, 255] \times [0, 255]$ following cases can occur,

- If $(\hat{x}_i, \hat{y}_i) \notin [0, 255] \times [0, 255]$, then the measurement is omitted.
- If $(\hat{x}_i, \hat{y}_i)$ coincides with a coordinate that belongs to a building, then the measurement is omitted since the position of buildings is known and is automatically set to zero in the measurement map.
- If $(\hat{x}_i, \hat{y}_i)$ coincides with $(\hat{x}_j, \hat{y}_j)$, then the measurement with lower indices is overwritten. E.g. if $i < j$, then measurement i will be overwritten.

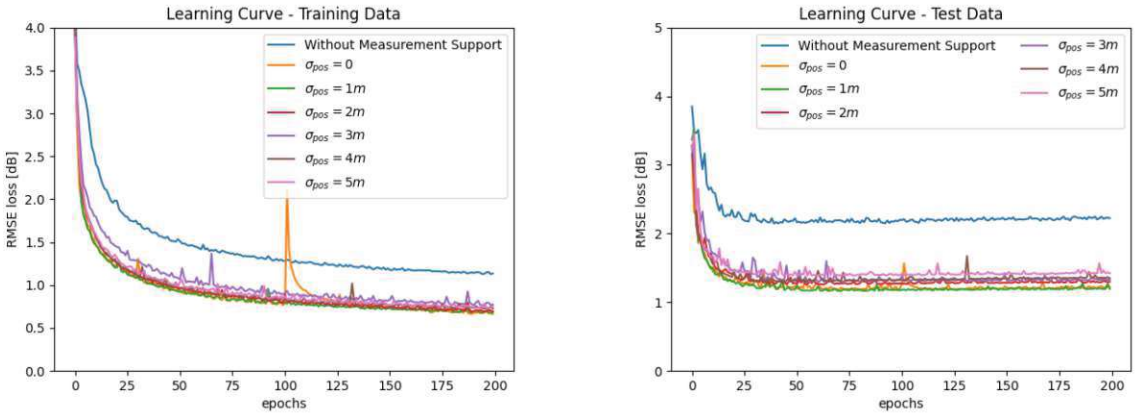
Similar to the last section (Sec. 5.1), the higher the position distribution is, the more information gets lost. The resulting NMD is necessarily smaller than the original map from $\mathcal{X}_{\text{Train},5\%}$ or $\mathcal{X}_{\text{Train},5\%}$. Again, this thesis considered them outliers and excluded them from the evaluation.

For the investigation the sensitivity of the network in dependence of the positional uncertainty, multiple training and test data set were created, each with different positional perturbation σ_{pos}^2, which are $\sigma_{\text{pos}}^2 \in \mathcal{N}_{\sigma_{\text{pos}}^2} = [1\text{m}^2, 4\text{m}^2, 9\text{m}^2, 16\text{m}^2, 25\text{m}^2]$. The process started with samples from $\mathcal{X}_{\text{Train},5\%}$ and $\mathcal{X}_{\text{Test},5\%}$. A method added positional perturbation with an intensity determined by σ_{pos}^2 to each measurement in order to generate the sets $\mathcal{X}_{\text{Train,P},\sigma^2}$ and $\mathcal{X}_{\text{Test,P},\sigma^2}$ for $\sigma_{\text{pos}}^2 \in \mathcal{N}_{\sigma_{\text{pos}}^2}$. The index P indicates the positional perturbation of each measurement.

5.2.2 Network Training

The network trained with training data sets with different σ_{pos}^2 to study the learning curves for different position measurement errors. Training parameters were $epchos = 200$ and $batch\ size = 16$. The learning curves for the training and test data sets and different position measurement uncertainty are shown in Fig. 5.6. The position perturbation has little effect on the training samples since all learning curves show similar behavior. The training loss curves cluster in such a way as to give the impression that the learning curves for training samples are almost insensitive to positional perturbations. The learning curves for the test samples show dependence on the position measurement uncertainty. This learning curve converges to higher loss when the perturbation is more dominant. However, for the selected range of σ_{pos}^2, the increase of the loss for increasing σ_{pos}^2 is not very pronounced. For $\sigma_{\text{pos}}^2 = 1\text{m}^2$, the learning curve is

similar to the learning curve of the unperturbed measurements. This observation might be because for $\sigma^2_{\text{pos}} = 1\text{m}^2$, the probability that a measuring position is distorted by at most one pixel is $p = 0.75$, and for a distortion by at most two pixels, it is $p = 0.97$. Therefore, it is most likely that the position of the measurement is off by two pixels or less from its original position. The signal strength near the original measurement position will likely be similar in magnitude to the original position. The deviation from the ground truth will be correspondingly unnoticeable. For larger σ^2_{pos}, the deviations from the ground truth slowly become dominant, resulting in a corresponding validation curve, which converges to a higher loss, as shown in Fig. 5.6b. However, even the learning curve for test samples with a distortion $\sigma^2_{\text{pos}} = 25\text{m}^2$, where the probability that measurements are off by at most 7 pixels from their original position is $p = 0.75$, converges to a loss that is closer to the loss of the undisturbed measurement than to the loss of a non-measurement supported UNet.

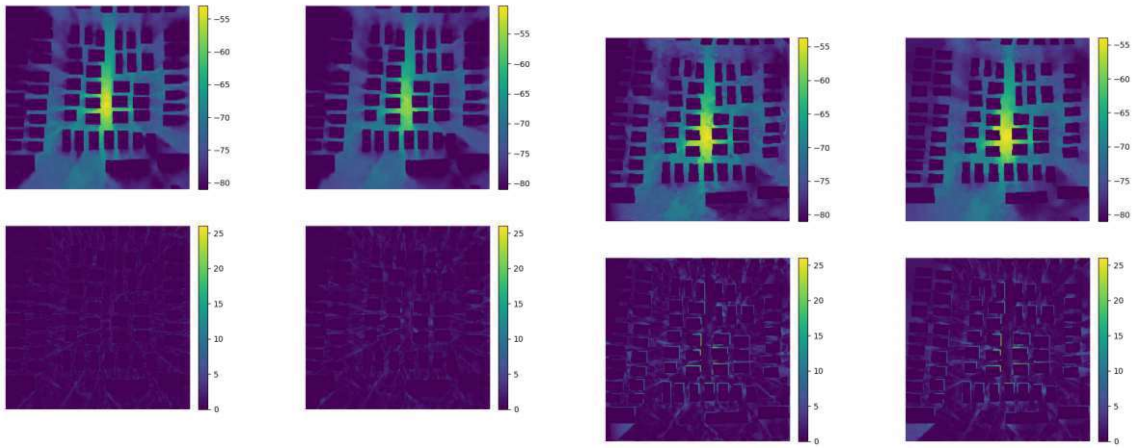


(a) Learning Curves for Training Data **(b)** Learning Curves for Test Data

Fig. 5.6: Learning Curves for different σ_{pos}

5.2.3 Prediction results

The results of the prediction of the known environment are shown in Fig. 2.6 with a position determination uncertainty of $\sigma^2_{\text{pos}} = 9\text{m}^2$. The UNets used in this section are an NMD-specialized UNet for MND = 5% and a UNet trained with distorted measurements with $\sigma^2_{\text{pos}} = 9\text{m}^2$, which is called *position noise specialized UNet*. Furthermore, comparing these and the baseline results generated by linear interpolation and KNN regression addresses the effect of introducing positional error to the network. Visual results are given in Fig. 5.7. The NMD-specialized UNET is also suitable for predicting this particular known sample. The error maps for both UNets show no noticeable deviation of the predictions from the ground truth. In contrast, the baselines show some odds in the shadows and deviations from the ground truth at the edge of the buildings.



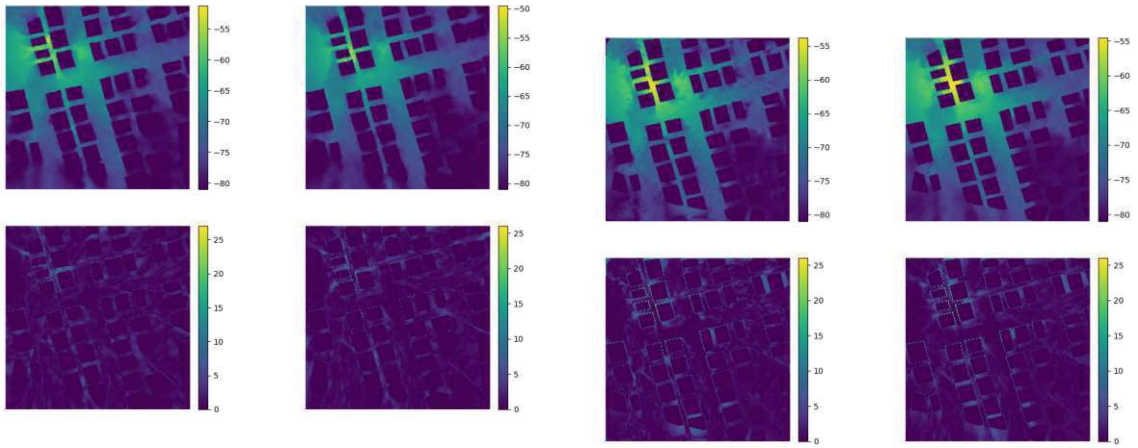
(a) Signal Noise Specialized UNet **(b)** NMD-Specialized UNet **(c)** Linear Interpolation **(d)** KNN Regression

Fig. 5.7: The first row shows the signal strength map for a known environment, that is generated with a position noise specialized UNet for $\sigma^2_{\text{pos}} = 9\text{m}^2$, NMD-specialized UNet for MND = 5%, linear interpolation and KNN regression. The second row shows the error map of each method.

The table below shows the RMSE for each method. As expected, the RMSE for a position noise specialized UNet produces a prediction with the smallest RMSE. The prediction from an NMD-specialized UNet has a slightly larger RMSE. Both methods outperform linear interpolation and KNN regression in accuracy.

	Position Noise Specialized UNet	NMD-Specialized UNet	Linear Interpolation	KNN Regression
RMSE	0.8352dB	1.1204dB	2.5198dB	2.6213dB

Figure 5.8 visually shows the predictions for an unknown environment, displayed in Fig. 2.7, with positionally distorted measurements with $\sigma^2_{\text{pos}} = 9\text{m}^2$, generated with the same UNets discussed above, and the baselines. The prediction error maps show that the signal strength predictions for both UNets and the baselines deviate from the ground truth in some areas, particularly around buildings.



(a) Signal Noise Specialized UNet **(b)** NMD-Specialized UNet **(c)** Linear Interpolation **(d)** KNN Regression

Fig. 5.8: The first row shows the signal strength map for an unknown environment, that is generated with a Signal Noise Specialized UNet for $\sigma^2 = 0.05$, NMD-specialized UNet for MND = 5%, linear interpolation and KNN regression. The second row shows the error map of each method.

The table below shows the RMSE for each method. Interestingly, the prediction of an NMD-specialized UNet has a smaller RMSE than the position noise specialized UNet, contrary to the expectation that the position noise specialized UNet should be able to handle positional perturbations. In order to investigate whether this is a coincidence, the RMSEs of all test samples of the test dataset with positionally distorted measurements with both UNets are computed and compared to the averaged RMSE.

	Position Noise Specialized UNet	NMD-Specialized UNet	Linear Interpolation	KNN Regression
RMSE	1.6989dB	1.6772dB	2.2636dB	2.2028dB

The average RMSEs over the standard deviation σ_{pos} are displayed in Fig. 5.9 for each method used in this section, which express the overall performance of each method. Figure 5.9 shows that the position noise specialized UNet trained has the smallest RMSE for the known environment over the entire chosen range of σ^2_{pos}. This UNet has learned to compensate for the uncertainty of the position measurement, which comparison with the NMD-specialized UNet shows. For an unknown environment, both networks' overall performance seems similar. Since the NMD-specialized UNet trained with undisturbed measurements performs slightly better than the position noise specialized UNet, it leads to the assumption that this UNet has lost the ability to counteract positional errors when it has to predict the signal strength of unknown environments. Numerical values are given in Tab. B.4 and B.5.

Linear Interpolation and KNN regression display high prediction inaccuracy with a significant variance. Numerical values are given in Tab. B.9. Like the last section, these methods rely on supporting points and observation. Therefore, they are more error-prone to positional measurement uncertainty.

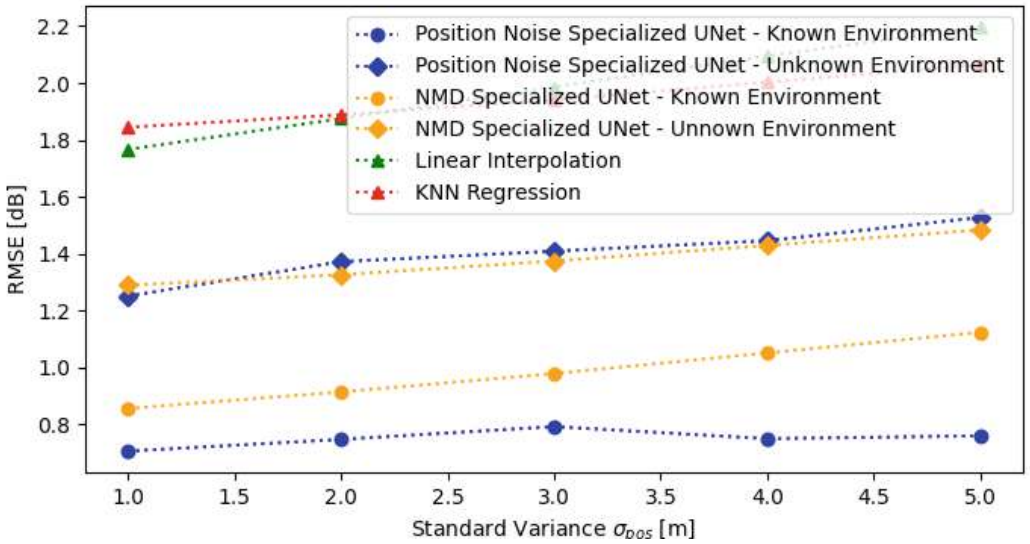


Fig. 5.9: Average RMSE and variance in dB scale over position uncertainty for the training dataset and test dataset and for different methods.

The end of this section focuses on the discussion of the R2 score of each method. Fig. 5.10 shows the results. For the known environment, the position noise specialized UNet has the highest R2 score over the chosen range of σ_{pos}, followed by the NMD-specialized UNet for MND = 5%, then linear interpolation, and KNN regression. Unlike Fig. 5.5, where the baseline generation models have negative R2 scores for some σ^2, in this analysis, the R2 score is always positive for the chosen range σ^2_{pos}. For the unknown environment, an unexpected property occurs, namely that the NMD-specialized UNet has a higher R2 score than the position noise specialized UNet, which should better handle positional distortions.

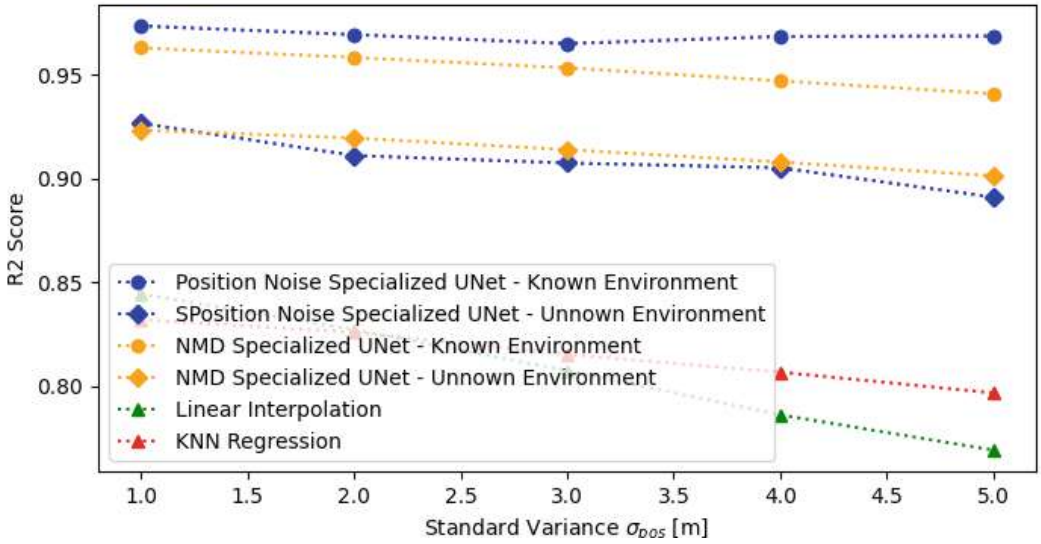


Fig. 5.10: Averaged R2 score over position uncertainty for different methods

5.3 Random Walker Measurement

Studies on the effect of the measurement channel on the UNet show that the predictions become more accurate when supported with many signal strength measurements. In this case, the measurements are distributed evenly over the entire map. However, it is rather unlikely that the measurements are distributed evenly over the street since, in some streets, the density of user devices, especially in busy areas, is higher than in other streets. Therefore, some parts of the map will obtain more information about the signal strength, while others have few measurements. A random walker that explores some streets while leaving others unscanned simulates this situation. This results in a measurement map with numerous signal strength measurements in some parts of the map, while it contains no information in other parts. In the previous simulation, a value NMD determined the number of measurements. The introduction of the value *coverage* links the number of measurements taken by the random walker and the NMD. The advantage of this value is the straightforward comparability. This value *coverage* describes the percentage of streets with signal strength measurement. The random walker explores the radio map until it reaches the given percentage of the area. The number of steps, and therefore the number of measurements, depends on the percentage of streets for a given urban scenario and coverage. The expression of the coverage value in percent directly relates to NMD.

5.3.1 Training and Test Dataset

Similar to Section 4.2.1, adding a measurement map to each sample of the training $\mathcal{X}_{\text{Train}}$ and test datasets $\mathcal{X}_{\text{Test}}$ resulted in training and test data sets with random walker measurements. The value *coverage* specified the distance and number of measurements. This value also determined

the percentage of the area to be measured. The random walker explored the map and measured until he achieved the specified street proportion. In order to analyze solely the effect of having information in some parts of the map, the measurements have to be accurate, i.e., there is no measurement error in the signal strength and the position where the measurement is. Each other regions on the measurement map that was not measured or did not have a building on them have a "no information" value, i.e., -1.

Now, this paragraph discusses the random walker. The first step is to set the starting point of the random walker to any position on the street area. Then, it can move one pixel in a horizontal, vertical, or diagonal direction, corresponding to 1m in the horizontal and vertical direction and $\sqrt{2}$m in a diagonal direction, which are eight directions in total. The movement of one pixel in any direction is called a step $\mathbf{s} = [s_x, s_y]$, where $s_x, s_y \in \{-1, 0, 1\}$, where $\mathbf{s} = [0, 0]$ is excluded. The first step will be in any direction, i.e., a step in each direction has the same probability. The current step is selected depending on its last step to ensure that the random walker has a preferred direction. A pmf for the current step conditioned on the last step expressed this property, i.e., the step $\mathbf{s}_k$ is drawn from the following pmf

$$\mathbf{s}_k \sim p(\mathbf{s}_k|\mathbf{s}_{k-1}). \tag{5.2}$$

The question arises about the structure of this pmf. Since it is desired that the current step be the last, the probability for $\mathbf{s}_k = \mathbf{s}_{k-1}$ must be much higher than in any other direction. Experiments with the following pmf

$$p(\mathbf{s}_k|\mathbf{s}_{k-1}) = \begin{cases} 0.72 & \text{for } \mathbf{s}_k = \mathbf{s}_{k-1} \\ 0.04 & \text{for one of the other seven directions} \end{cases}. \tag{5.3}$$

deliver good results. Therefore, this simulation used this pmf for the random walker. Other choices of the probabilities should also work, especially individual probabilities for each direction, as long as $p(\mathbf{s}_k = \mathbf{s}_{k-1}|\mathbf{s}_{k-1}) > p(\mathbf{s}_k \neq \mathbf{s}_{k-1}|\mathbf{s}_{k-1})$. For the choice made in this simulation the probability is $p(\mathbf{s}_k = \mathbf{s}_{k-1}|\mathbf{s}_{k-1}) = 0.72 > p(\mathbf{s}_k \neq \mathbf{s}_{k-1}|\mathbf{s}_{k-1}) = 0.28$.

The random walker cannot enter buildings. For example, a building area will choose another direction if the next step is a building area. It also cannot leave the map. If the next step leaves it, the random walker moves in another direction.

Due to the definition of *coverage* in this thesis, there is a direct relation between NMD and *coverage*[1]. In Sec. 4.2.1[2], the number of measurements is within the set $\mathcal{N}_{\text{NMD}} = \{1\%, 2\%, 5\%, 10\%, 20\%\}$. To compare how the UNet performs with a similar amount of measurements, training, and test data sets were created for each coverage $\in \mathcal{N}_{\text{coverage}} = \{1\%, 2\%, 5\%, 10\%, 20\%\}$. Examples of measurement maps with different *coverage* for a given urban scenario and antenna placement are in Fig. 5.11.

[1] Note that for measurements spread over the entire map, the positions of the measurements, and thus the total number of measurements, are random. In this case, the proportion of measured street areas is an expected value.

[2] This section considers uniformly distributed measurements over the whole map

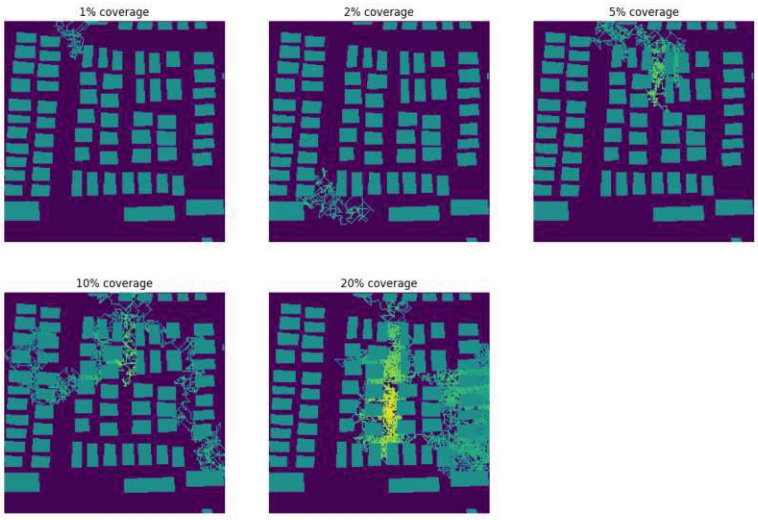


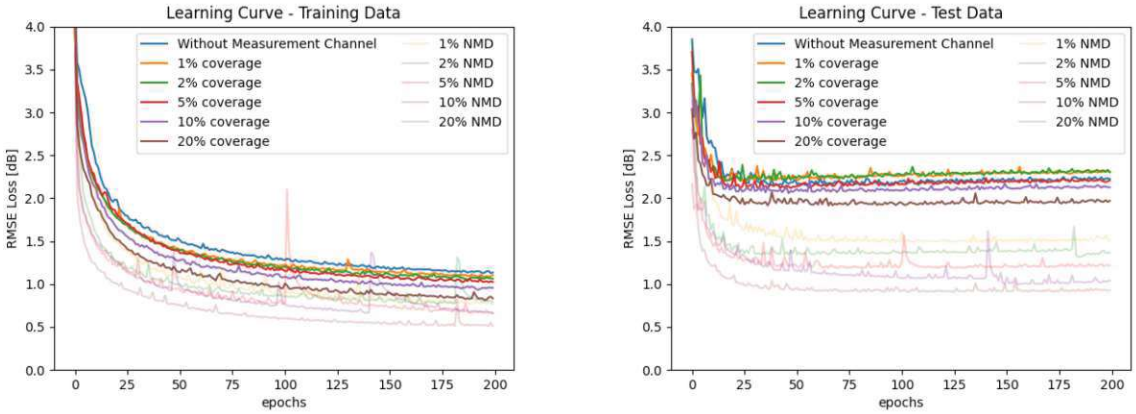
Fig. 5.11: Measurement maps with different *coverage* for a chosen urban scenario and antenna placement

5.3.2 Network Training

For each coverage $\in \mathcal{N}_{\text{coverage}}$, a UNet was trained with 200 epochs and a batch size of 16. This UNet is called *random walker specialized UNet*. The results of each training process are shown in Fig. 5.12 as opaque lines. For the training dataset, Fig. 5.12a shows that the learning curve converges to more minor losses when the random walker explores more extensive regions. However, Fig. 5.12b shows that for coverage below 5%, the learning curve converges to a more significant loss than the learning curve of non-measurement supported UNet. The additional information in the test samples of these test data sets is likely to confuse the neural network, resulting in a higher loss. In this case, the neural network should ignore the measurement map and perform at least as well as the two-channel UNet. When the coverage increases, the converged loss decreases. If the random walker jas measured more than 5% of the street area, then the loss is smaller than that of the non-measurement supported UNet.

For comparison, the learning curves of the network training and validation with measurements spread over the whole street area from Section 4.1.4 are shown in semitransparent colors in Fig. 5.12. Note that NMD-Specialized UNet was trained and validated with training and test datasets that contain measurement maps with measurements distributed over the entire map, i.e., $\mathcal{X}_{\text{train},NMD}$ and $\mathcal{X}_{\text{test},NMD}$. For training data, the NMD Specialized UNet and the random walker specialized UNet behave very similarly, with the learning curves of the random walker specialized UNet ending with a higher loss after training. The learning curves for validating the random walker specialized UNet have exciting properties compared to those of the NMD-Specialized UNet. First, the validation loss converges before 25 epochs, earlier than the validation loss of the NMD-Specialized UNet, which converges at about 50 epochs. Furthermore, the validation losses

of the random walker specialized UNet are more significant than those of the NMD-Specialized UNet for any coverage. Furthermore, the validation loss for an NMD-Specialized UNet for MND = 1% is much smaller than that for a random walker specialized UNet for coverage = 20%. These losses are visible in Fig. 5.12b.

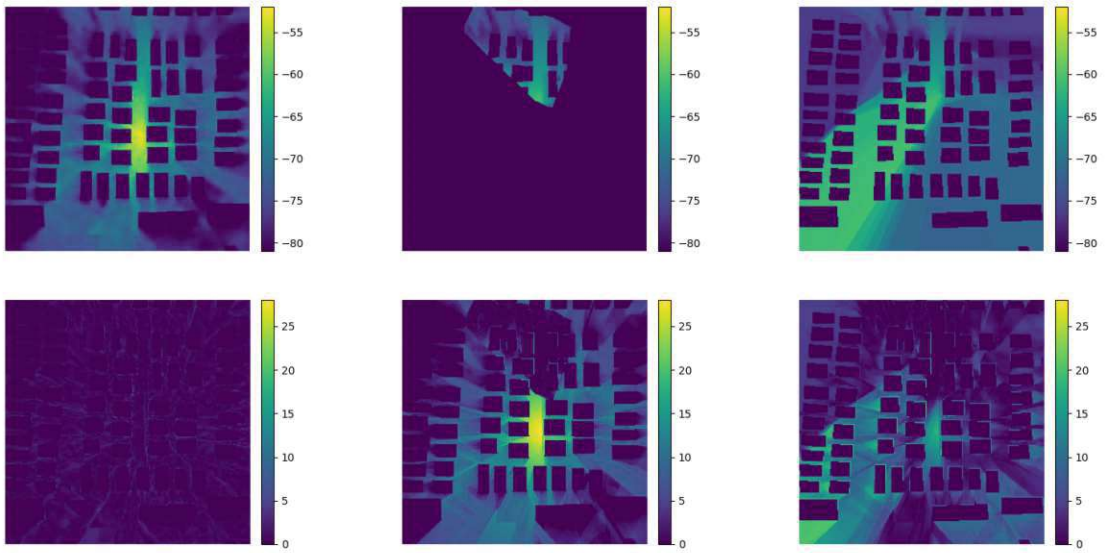


(a) Learning Curves for Training Data **(b)** Learning Curves for Test Data

Fig. 5.12: Learning Curves for various *steps*

5.3.3 Prediction Results

Fig. 2.6 displays the urban environment of the training sample for evaluating the prediction of the UNet. Passing this urban environment along with a measurement map with coverage = 5% through a random walker specialized UNet for coverage = 5% results in a prediction, shown in Fig. 5.13a The signal strength structure and the buildings' shadowing in this prediction are detailed. Accordingly, the error map does not display any spot with a significant deviation from the ground truth. The Fig. 5.13b and 5.13c show the baselines generated by linear interpolation and KNN regression. It follows that the UNet can predict the complete map even if parts of the maps are unknown. At the same time, the interpolation and regression generate only parts of the map that make sense, especially where the random walker has made some measurements. Regions away from the measurements have no supporting points or observations, so the interpolated values go to the noise level. At the same time, the KNN regressor selects the closest observations for a given point and averages the observation values. The error maps corresponding to these methods show low errors around the area measured and considerable deviations from ground truth everywhere else.



(a) UNet Prediction **(b)** Linear Interpolation **(c)** KNN Regression

Fig. 5.13: The first row shows the signal strength map that is generated with a UNet, linear interpolation, and KNN regression for a training sample. The second row shows the error map of each method.

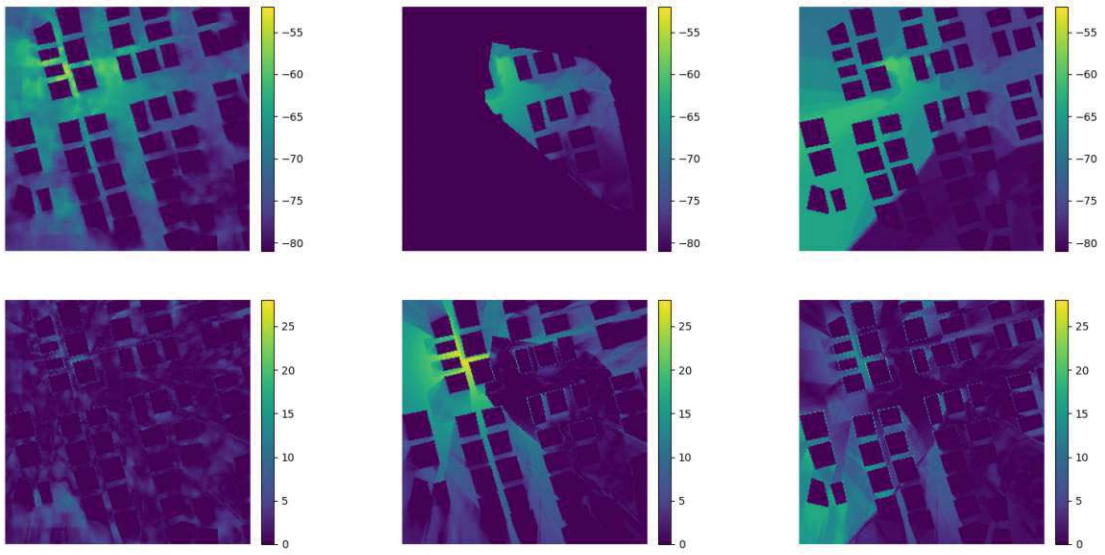
The RMSEs of the prediction and baseline for this particular test sample for each method are

	UNet	Linear Interpolation	KNN Regression
RMSE	1.0102dB	7.7519dB	7.3897dB

While the UNet can produce predictions with comparatively low RMSE, linear interpolation, and KNN regression have problems with limited information, and they tend to have larger RMSE for these two methods, whereby the RMSE of the KNN regression baseline shows a slightly smaller RMSE than the linear interpolation baseline. This UNet cannot compete with the NMD-Specialized UNet for MND = 5%. The prediction of this network with a sample of the same urban environment but with measurements uniformly distributed over the entire map has an RMSE of RMSE = 0.7837dB. However, in both cases, the measurement maps have a similar number of measurements.

Next, a test sample shown in Fig. 2.7 was passed through the network with a measurement map with coverage = 5%. The prediction is shown in Fig. 5.14a. Compared to baselines generated by linear interpolation or KNN regression in Fig. 5.14b and 5.14c, the UNet can generate a complete signal strength prediction map with comparatively lower RMSE. However, the prediction shows a blurred distribution of the signal strength and the shadows cast by the building. This results in a larger region with more deviation from the ground truth. Problems with the baselines generated

by linear interpolation and KNN are the same as for the training sample, and they also apply here.



(a) UNet Prediction **(b)** Linear Interpolation **(c)** KNN Regression

Fig. 5.14: The first row shows the signal strength map that is generated with a UNet, linear interpolation, and KNN regression for a test sample. The second row shows the error map of each method.

The RMSEs of the prediction and baseline for this particular test sample for each method are

	UNet	Linear Interpolation	KNN Regression
RMSE	2.5498dB	6.5061dB	5.0632dB

For these samples with random walker measurements, the RMSE is larger compared to the prediction with distributed measurements. For comparison, a NMD-specialized UNet predicts RMSE = 1.5468dB for the same urban environment but with distributed measurements with a MND = 5% measurement density. The result of the random walker specialized UNet is comparable to the two-channel UNet without measurement support, generating a prediction with RMSE = 2.8347dB. Although the RMSE of the baselines for this test sample is smaller for the training sample, they do not depend on either the training or the test data sample but on the path of the random walker, which can lead to significant variation of the RMSE compared to the baselines of a training sample.

The computation for averaged RMSE of the predictions for the training and test datasets took place separately and for each coverage $\in \mathcal{N}_{\text{coverage}}$ to obtain the overall performance of the

random walker specialized UNet in dependence of the coverage. The results are shown in Fig. 5.15 along with the averaged RMSE of the NMD-Specialized UNet and for the baselines. The statistics of each RMSE are also shown in this figure to emphasize the considerable variation in the accuracy of the baselines.

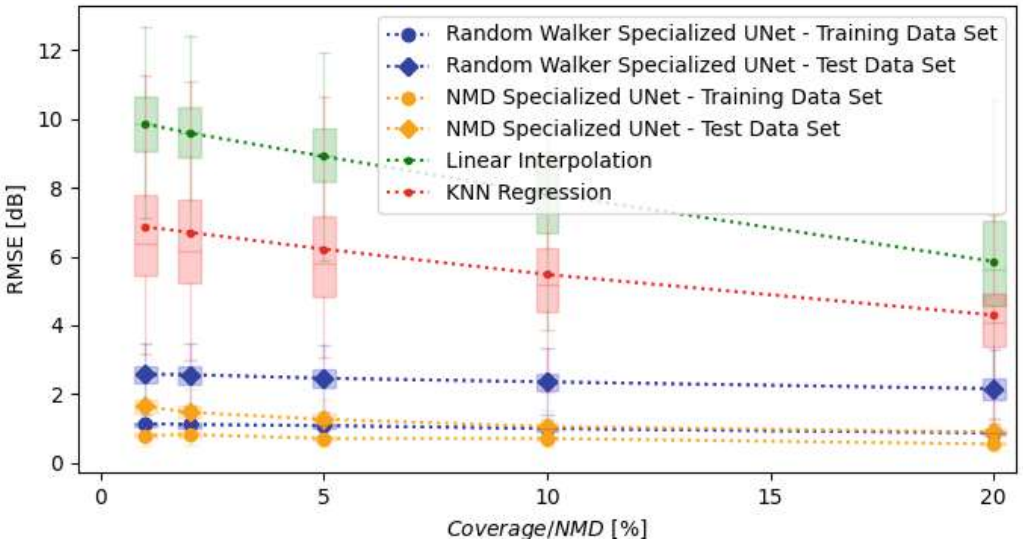


Fig. 5.15: Average RMSE and variance in dB scale for all training and test datasets samples.

The baselines have the largest averaged RMSE over the entire defined range, with the averaged RMSE of the linear interpolator being more significant than that of the KNN regressor. The averaged RMSE of these methods decreases as larger areas are measured, i.e., the random walker takes more steps and explores larger areas. Furthermore, these methods have the most significant variation. The accuracy of the baselines will suffer if the random walker starts at a position with low signal strength and explores only tiny regions with a given number of steps or regions where the signal strength is generally low. Exploring large areas with the same number of steps and areas around the antenna provides more valuable information about the signal strength, resulting in more accurate baselines. Since the starting point of the random walker is random, the datasets indeed include these cases in the datasets. The random walker specialized UNet exhibits smaller averaged RMSE than the linear interpolation and KNN regression. The network has higher prediction accuracy for samples in the training data set than those in the test data sample. There is a slight and barely noticeable decrease in the averaged RMSE with increasing coverage, implying that the prediction accuracy does not improve on average as the random walker explores large areas. In addition, the variance of the individual RMSEs is not as significant as for the baselines. However, Fig. 5.15 also shows superior performance for uniformly distributed measurements.

Finally, the section discusses the R2 score for random walker specialized UNet, NMD Specialized UNet, linear interpolation, and KNN regression. The R2 score for each method is given in Fig. 5.16, whereby for UNets, the R2 score is distinguished between training and test samples. The

R2 scores for the corresponding training dataset of both UNets show a value close to one over the defined coverage or NMD range. For NMD-Specialized UNet, the R2 score is close to one for the test dataset. In contrast, the random walker specialized measurement has a lower R2 score for test samples. There is a slight increase of these R2 scores by increasing the coverage or the NMD.

For the baseline generated by linear interpolation and KNN regression, R2 scores show negative values even for extensive coverage. The negative score is apparent since large regions of the signal strength prediction map are not reconstructible if the random walker does not traverse around these regions. More extensive coverage increases the probability of exploring more extended regions so that a larger region can be interpolated or regressed. The higher exploration leads to an increasing R2 score, but as shown in Fig. 5.16, the R2 score is almost always below zero for the given range.

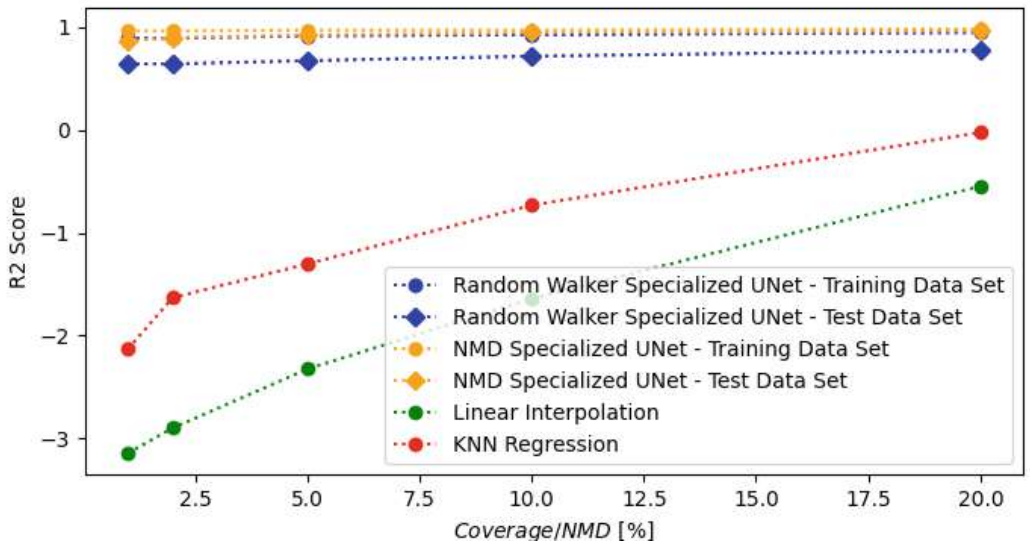


Fig. 5.16: Averaged R2 score for different methods

5.4 Summary

- Real-world measurements will always suffer from measurement uncertainties. Therefore, simulation has been done by adding power signal noise and position error to the signal power measurement.
- A UNet can be trained with these measurement uncertainties to learn how to handle and counteract erroneous measurements.
- Baselines are heavily affected by measurement errors.
- A more realistic scenario is that the measurements are concentrated in a few regions. Simulation this scenario results in a measurement maps with much information, while other

parts have no information about signal strength. This scenario will be simulated with a random walker. It explores the map until the defined ratio of the street area is measured.

- A UNet can predict the signal strength outside the measured area more accurately than the baselines generated by linear interpolation and KNN regression.
- For the highest prediction accuracy, it is necessary to have some information in every region of a given urban environment.

Chapter 6

Conclusion

This thesis examined various techniques to enhance UNet models for RME, focusing on introducing a measurement channel. The measurement channel is the most effective approach, providing the network with additional signal strength information and significantly improving prediction accuracy.

For UNets without a measurement channel, the prediction quality varies depending on the characteristics of the city maps. Predictions for areas with a high proportion of roads typically show higher accuracy. At the same time, in densely built-up environments, the UNet must account for complex propagation characteristics, resulting in increased RMSE. However, the prediction accuracy in certain road-dominant areas is below expectations, underscoring the necessity for a calibration mechanism. The network utilizes partial signal strength data to enhance predictions by incorporating a measurement channel. Experiments have shown that a distributed measurement over the complete map with a density of approximately 0.05 measurements per square meter significantly improves the prediction accuracy by 1.2054dB for unknown urban environments.

The study also examined partially explored city map scenarios and the inclusion of measurement errors to ensure a more realistic representation of real-world conditions. UNets can compensate for measurement errors, even for distortions of 12.9 dB. When only an incomplete city map scan is available, the UNet relied on its model for predictions in areas lacking measurements, which increased RMSE compared to distributed measurements. However, the UNet significantly outperformed baseline methods, which are restricted to predicting signal strength only in areas close to the measurements and assessing the accuracy of those measurements. These findings highlight the advantage of incorporating at least some information across the entire area of interest and the method's resilience against measurement distortions, regardless of the approach employed.

For real-world applications, the UNet requires further adaptations. These include training the UNet with a broader range of urban environments that account for elevation, diverse terrains, and multiple carrier frequencies. One of the main challenges is designing a UNet that can learn propagation mechanics from incomplete scans of the urban environment, enabling accurate predictions for unknown city maps. Reducing the dependency on fully detailed radio maps lessens the effort to generate ground truth data.

A critical aspect of practical implementation is assessing prediction reliability. This thesis proposes integrating uncertainty measures, including epistemic uncertainty (arising from

model limitations) and aleatoric uncertainty (stemming from input variability) [38], into the network. These indicators provide valuable insights into prediction confidence, enabling targeted adjustments in low-confidence areas through additional simulations or measurements.

In summary, this thesis presents a robust and efficient framework for RME using deep learning. The findings indicate that including calibration measurements significantly enhances prediction accuracy. Additionally, UNet-based models provide reliable predictions, even in realistic scenarios that involve incomplete data and measurement noise. These advancements open new possibilities for resource-efficient network planning, optimization, and real-world implementation, paving the way for future developments in telecommunication network management.

Appendices

Appendix A

Implementation

A.1 Measurement Map

The algorithm A.1 shows how to generate the measurement map $\boldsymbol{X}_{\text{measurement}}$. First, an array is created with $N_1 \times N_2 = 256 \times 256$ elements, which are random values, independently and identically distributed according to a Gaussian distribution with limits from 0 to 1. Any element that is $1 - p$, where p corresponds to the NMD in this project, will be set to -1, and all others will be equal to the ground truth. Since we know the position of the buildings and are not interested in the signal strength inside the buildings, all elements are set to 0, i.e., $\boldsymbol{X}_{ij} = 0$.

Algorithm A.1: Generating Measurement Map

Data: -

Result: Measurement Map $\boldsymbol{X}_{\text{measurement}}$

Create an array $\boldsymbol{X}_{\text{measurement}}$ of size $N_1 \times N_2$;
Assign each element a random value according to $\boldsymbol{X}_{ij} \sim \mathcal{U}(0, 1)$;
Set each element $\boldsymbol{X}_{ij} < 1 - p$ to $\boldsymbol{X}_{ij} = -1$;
Set each element $\boldsymbol{X}_{ij} \geq p$ to $\boldsymbol{X}_{ij} = y_{ij}$;
for $\forall$ *elements* $\boldsymbol{X}_{ij}$ *of* $\boldsymbol{X}_{measurement}$ **do**
    if *Index* i, j *corresponds to building area* **then**
        Set $\boldsymbol{X}_{ij} = 0$;
    end
end

A.2 Linear Interpolation

In the first step, we extract all elements $\boldsymbol{X}_{ij}$ of $\boldsymbol{X}_{\text{measurement}}$ that corresponds to a measured position, i.e., $\boldsymbol{X}_{ij} \neq -1$ and the indices i, j do not correspond to the building area. These elements form the support points for the linear interpolation $\boldsymbol{P}_{\text{lin.int.}}$. After generating $\boldsymbol{P}_{\text{lin.int.}}$ we set all values $\boldsymbol{P}_{ij}$ whose indices i, j belong to the building area to zero, i.e. $\boldsymbol{P}_{ij} = 0$.

Experiments have shown that if we add the information about the buildings to the support points, which is easy to do since the signal strength is set to zero, the accuracy of the baseline will decrease. The problem is that the buildings drag the surrounding signal strength to a lower value due to the linear interpolation.

Algorithm A.2: Liner Interpolation Baseline

Data: Measurement Map $\boldsymbol{X}_{\text{measurement}}$

Result: Linear interpolation Baseline $\boldsymbol{P}_{\text{lin.int.}}$

create an empty list *support points*;
for $\forall$ *elements* $\boldsymbol{X}_{ij}$ *of* $\boldsymbol{X}_{measurement}$ **do**
  if *Index* i, j *corresponds to a measured position* **then**
    Add $([i, j], \boldsymbol{X}_{ij})$ to list *support points* ;
  end
end
Call `LinearNDInterpolator` class from `scipy` package;
Pass *support points* to `LinearNDInterpolator` class;
Perform piece wise linear interpolation $\boldsymbol{P}_{\text{lin.int.}}$;
for $\forall$ *elements* $\boldsymbol{P}_{ij}$ *of* $\boldsymbol{P}_{lin.int.}$ **do**
  if *Index* i, j *corresponds to building area* **then**
    Set $\boldsymbol{P}_{ij} = 0$
  end
end

A.3 KNN Regression

The procedure is similar to linear interpolation. Each element $\boldsymbol{X}_{ij}$ of the measurement map $\boldsymbol{X}_{\text{measurement}}$ corresponding to a measurement is collected in a list called *observation*. Then, with this information, a KNN regression is performed, creating a baseline $\boldsymbol{P}_{\text{knn.reg.}}$. Then all elements $\boldsymbol{P}_{ij}$ whose indices i, j belong to the building area are set to zero, i.e. $\boldsymbol{P}_{ij} = 0$.

Algorithm A.3: KNN Regression Baseline

Data: Measurement Map $\boldsymbol{X}_{\text{measurement}}$

Result: KNN Regression Baseline $\boldsymbol{P}_{\text{knn.reg.}}$

create an empty list *observation*;
for $\forall$ *elements* $\boldsymbol{X}_{ij}$ *of* $\boldsymbol{X}_{measurement}$ **do**
  if *Index* i, j *corresponds to a measured position* **then**
    Add $([i, j], \boldsymbol{X}_{ij})$ to list *observation* ;
  end
end
Call `KNeighborsRegressor` class from `scipy` package;
Pass *observation* to `KNeighborsRegressor` class;
Perform KNN regression with `n_neighbors=5` and obtain $\boldsymbol{P}_{\text{knn.reg.}}$;
for $\forall$ *elements* $\boldsymbol{P}_{ij}$ *of* $\boldsymbol{P}_{knn.reg.}$ **do**
  if *Index* i, j *corresponds to building area* **then**
    Set $\boldsymbol{P}_{ij} = 0$
  end
end

Appendix B

Tables

B.1 Tables for Normalized Measurement Density Analysis

Tab. B.1: Average RMSE over all samples from the training and test dataset along with the standard deviation for different NMD for UNet prediction.

NMD	Average RMSE Training Data Sample	Std. Dev. of RMSE Training Data Sample	Average RMSE Test Data Sample	Std. Dev. of RMSE Test Data Sample
1%	0.7968dB	0.1044dB	1.6438dB	0.3092dB
2%	0.8229dB	0.1180dB	1.4683dB	0.2658dB
5%	0.7099dB	0.1000dB	1.2693dB	0.2420dB
10%	0.7108dB	0.0940dB	1.0578dB	0.1900dB
20%	0.5478dB	0.0796dB	0.8972dB	0.1742dB

Tab. B.2: Average RMSE and standard deviation for different NMD for linear interpolation and KNN regression.

NMD	Average RMSE Linear Interpolation	Std. Dev. of RMSE Linear Interpolation	Average RMSE KNN Regression	Std. Dev. of RMSE KNN Regression
1%	2.3204dB	0.3244dB	2.3161dB	0.4127dB
2%	1.9887dB	0.3279dB	2.0692dB	0.3944dB
5%	1.6829dB	0.3199dB	1.8168dB	0.3666dB
10%	1.6829dB	0.3199dB	1.8168dB	0.3666dB
20%	1.4214dB	0.3124dB	1.5537dB	0.3342dB

Tab. B.3: Average RMSE over all prediction of the NMD-flexible UNet for known and unknown environments

NMD	Average RMSE known Env.	Std. Dev. of RMSE known Env.	Average RMSE unknown Env.	Std. Dev. of RMSE unknown Env.
1%	1.7767dB	0.3293dB	2.0274dB	0.3727dB
2%	1.2691dB	0.2214dB	1.5666dB	0.2889dB
5%	1.0238dB	0.1644dB	1.3042dB	0.2384dB
10%	1.0125dB	0.1623dB	1.2279dB	0.2200dB
20%	1.2090dB	0.1651dB	1.3236dB	0.2043dB

B.2 Tables for Singal Noise Analysis

Tab. B.4: Average RMSE and Std. Dev. for signal noise specialized UNet prediction for known and unknown environments with different standard deviation of the noise σ

σ	Average RMSE Known Environment	Std. Dev. of RMSE Known Environment	Average RMSE Unknown Environment	Std. Dev. of RMSE Unknown Environment
2.9dB	0.7588dB	0.0959dB	1.5616dB	0.2634dB
4.1dB	0.8334dB	0.0982dB	1.7822dB	0.2825dB
6.48dB	0.8534dB	0.1073dB	1.9665dB	0.2822dB
9.17dB	0.8605dB	0.1036dB	2.1148dB	0.2689dB
12.97dB	0.8867dB	0.0973dB	2.3465dB	0.2839dB

Tab. B.5: Average RMSE and Std. Dev. for NMD-specialized UNet prediction, trained with $NMD = 5\%$, for known and unknown environments with different standard deviation of the noise σ

σ	Average RMSE Known Environment	Std. Dev. of RMSE Known Environment	Average RMSE Unknown Environment	Std. Dev. of RMSE Unknown Environment
2.9dB	1.3780dB	0.1156dB	1.7228dB	0.2019dB
4.1dB	2.0122dB	0.1756dB	2.2508dB	0.2160dB
6.48dB	3.5330dB	0.2910dB	3.6376dB	0.2894dB
9.17dB	5.4150dB	0.3769dB	5.4644dB	0.3746dB
12.97dB	8.0887dB	0.4750dB	8.0856dB	0.4732dB

Tab. B.6: Average RMSE and Std. Dev. for linear interpolation and KNN regression for samples with different σ

σ	Average RMSE Linear Interpolation	Std. Dev. of RMSE Linear Interpolation	Average RMSE KNN Regression	Std. Dev. of RMSE KNN Regression
2.9dB	2.6402dB	0.2011dB	2.2478dB	0.2972dB
4.1dB	3.3343dB	0.1821dB	2.6990dB	0.2997dB
6.48dB	4.8081dB	0.2163dB	3.8079dB	0.3634dB
9.17dB	6.6186dB	0.3141dB	5.3584dB	0.4647dB
12.97dB	9.4374dB	0.4255dB	7.9084dB	0.5802dB

B.3 Tables for Position Noise Analysis

Tab. B.7: Average RMSE and Std. Dev. for position noise specialized UNet prediction for known and unknown environments with different standard deviation of the position noise σ_{pos}

σ_{pos}	Average RMSE Known Environment	Std. Dev. of RMSE Known Environment	Average RMSE Unknown Environment	Std. Dev. of RMSE Unknown Environment
1m	0.7061dB	0.1038dB	1.2513dB	0.2348dB
2m	0.7474dB	0.1057dB	1.3720dB	0.2511dB
3m	0.7923dB	0.1118dB	1.4090dB	0.2658dB
4m	0.7499dB	0.1054dB	1.4457dB	0.2710dB
5m	0.7600dB	0.1058dB	1.5277dB	0.2786dB

Tab. B.8: Average RMSE and Std. Dev. for NMD-specialized UNet prediction, trained with $NMD = 5\%$, for known and unknown environments with different standard deviation of the position noise σ_{pos}

σ_{pos}	Average RMSE Known Environment	Std. Dev. of RMSE Known Environment	Average RMSE Unknown Environment	Std. Dev. of RMSE Unknown Environment
1m	0.8561dB	0.1352dB	1.2893dB	0.2474dB
2m	0.9141dB	0.1513dB	1.3260dB	0.2575dB
3m	0.9789dB	0.1680dB	1.3743dB	0.2692dB
4m	1.0512dB	0.1867dB	1.4291dB	0.2841dB
5m	1.1238dB	0.2043dB	1.4834dB	0.2906dB

Tab. B.9: Average RMSE over all samples from the training and test dataset along with the Std. Dev. for different σ_{pos} for linear interpolation and KNN regression

σ_{pos}	Average RMSE Linear Interpolation	Std. Dev. of RMSE Linear Interpolation	Average RMSE KNN Regression	Std. Dev. of RMSE KNN Regression
1m	1.7664dB	0.3199dB	1.8444dB	0.3716dB
2m	1.8749dB	0.3258dB	1.8880dB	0.3777dB
3m	1.9862dB	0.3373dB	1.9408dB	0.3841dB
4m	2.0945dB	0.3429dB	2.0027dB	0.3882dB
5m	2.1970dB	0.3565dB	2.0637dB	0.3921dB

B.4 Tables for Random Walker Analysis

Tab. B.10: Average RMSE over all samples from the training and test dataset along with the Std. Dev. for different *coverage* for UNet prediction

coverage	Average RMSE Training Data Sample	Std. Dev. of RMSE Training Data Sample	Average RMSE Test Data Sample	Std. Dev. of RMSE Test Data Sample
1%	1.1397dB	0.1719dB	2.5771dB	0.3633dB
2%	1.1167dB	0.1863dB	2.5654dB	0.3934dB
5%	1.0832dB	0.1375dB	2.4613dB	0.3461dB
10%	0.9994dB	0.1303dB	2.3560dB	0.3919dB
20%	0.8738dB	0.1121dB	2.1579dB	0.4353dB

Tab. B.11: Average RMSE over all samples from a subset training and test dataset along with the Std. Dev. for different coverage for linear interpolation and KNN regression

coverage	Average RMSE Linear Interpolation	Std. Dev. of RMSE Linear Interpolation	Average RMSE KNN Regression	Std. Dev. of RMSE KNN Regression
1%	9.8566dB	1.0738dB	6.8641dB	2.1271dB
2%	9.5934dB	1.0877dB	6.7032dB	2.1422dB
5%	8.9126dB	1.1997dB	6.2165dB	1.8558dB
10%	7.8245dB	1.5482dB	5.4789dB	1.5833dB
20%	5.8560dB	1.6661dB	4.2985dB	1.3326dB

Acronyms

(R)MSE (root) mean squared error

CNN Convolutional Neural Network

CS crowd sensing

GPS Global Positioning System

KNN k-nearest neighbors

NMD normalized measurement density

pmf probability mass function

RME Radio Map Estimation

Bibliography

[1] W. El-Beaino, A. M. El-Hajj, and Z. Dawy, "On radio network planning for next generation 5G networks: A case study", in *2015 International Conference on Communications, Signal Processing, and their Applications (ICCSPA'15)*, 2015, pp. 1–6. DOI: 10.1109/ICCSPA.2015.7081315.

[2] D. Romero and S.-J. Kim, "Radio map estimation: A data-driven approach to spectrum cartography", *IEEE Signal Processing Magazine*, vol. 39, no. 6, pp. 53–72, 2022. DOI: 10.1109/MSP.2022.3200175.

[3] Z. Yun and M. F. Iskander, "Ray tracing for radio propagation modeling: Principles and applications", *IEEE Access*, vol. 3, pp. 1089–1100, 2015. DOI: 10.1109/ACCESS.2015.2453991.

[4] X. Zhang, X. Shu, B. Zhang, J. Ren, L. Zhou, and X. Chen, "Cellular network radio propagation modeling with deep convolutional neural networks", *CoRR*, vol. abs/2110.01848, 2021. arXiv: 2110.01848. [Online]. Available: https://arxiv.org/abs/2110.01848.

[5] Ç. Yapar, F. Jaensch, R. Levie, G. Kutyniok, and G. Caire, "Overview of the first pathloss radio map prediction challenge", *IEEE Open Journal of Signal Processing*, vol. 5, pp. 948–963, 2024. DOI: 10.1109/OJSP.2024.3419563.

[6] M. Hata, "Empirical formula for propagation loss in land mobile radio services", *IEEE Transactions on Vehicular Technology*, vol. 29, no. 3, pp. 317–325, 1980. DOI: 10.1109/T-VT.1980.23859.

[7] E. Kommission and G. I. und Medien, *COST Action 231 : Digital mobile radio towards future generation systems: Final Report*. Publications Office, 1999.

[8] R. Levie, Ç. Yapar, G. Kutyniok, and G. Caire, "RadioUNet: Fast radio map estimation with convolutional neural networks", *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 4001–4015, 2021. DOI: 10.1109/TWC.2021.3054977.

[9] C. Yapar, R. Levie, G. Kutyniok, and G. Caire, *Dataset of pathloss and toa radio maps with localization application*, 2022. DOI: 10.21227/0gtx-6v30. [Online]. Available: https://dx.doi.org/10.21227/0gtx-6v30.

[10] W. Wang, D. Qin, S. Wang, Y. Fang, and Y. Zheng, "A multi-channel UNet framework based on snmf-dcnn for robust heart-lung-sound separation", *Computers in Biology and Medicine*, vol. 164, p. 107 282, 2023, ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2023.107282. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0010482523007473.

[11] M. Molinari, M.-R. Fida, M. K. Marina, and A. Pescape, "Spatial interpolation based cellular coverage prediction with crowdsourced measurements", in *Proceedings of the 2015 ACM SIGCOMM Workshop on Crowdsourcing and Crowdsharing of Big (Internet) Data*, ser. C2B(1)D '15, London, United Kingdom: Association for Computing Machinery, 2015, pp. 33–38, ISBN: 9781450335393. DOI: 10.1145/2787394.2787395. [Online]. Available: https://doi.org/10.1145/2787394.2787395.

[12] S. Tripkovic, P. Svoboda, V. Raida, and M. Rupp, "Cluster density in crowdsourced mobile network measurements", in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–7. DOI: 10.1109/VTC2021-Spring51267.2021.9448661.

[13] J. Riihijarvi and P. Mahonen, "Machine learning for performance prediction in mobile cellular networks", *IEEE Computational Intelligence Magazine*, vol. 13, no. 1, pp. 51–60, 2018. DOI: 10.1109/MCI.2017.2773824.

[14] V. V. Ratnam, H. Chen, S. Pawar, *et al.*, "Fadenet: Deep learning-based mm-wave large-scale channel fading prediction and its applications", *IEEE Access*, vol. 9, pp. 3278–3290, 2021. DOI: 10.1109/ACCESS.2020.3048583.

[15] L. Eller, P. Svoboda, and M. Rupp, "A deep learning network planner: Propagation modeling using real-world measurements and a 3D city model", *IEEE Access*, vol. 10, pp. 122 182–122 196, 2022. DOI: 10.1109/ACCESS.2022.3223097.

[16] S. Shrestha, X. Fu, and M. Hong, "Deep spectrum cartography: Completing radio map tensors using learned neural models", *IEEE Transactions on Signal Processing*, vol. 70, pp. 1170–1184, 2022, ISSN: 1941-0476. DOI: 10.1109/tsp.2022.3145190. [Online]. Available: http://dx.doi.org/10.1109/TSP.2022.3145190.

[17] A. Chaves-Villota and C. A. Viteri-Mera, "Deeprem: Deep-learning-based radio environment map estimation from sparse measurements", *IEEE Access*, vol. 11, pp. 48 697–48 714, 2023. DOI: 10.1109/ACCESS.2023.3277248.

[18] S. Hong, H. Noh, and B. Han, *Decoupled deep neural network for semi-supervised semantic segmentation*, 2015. arXiv: 1506.04924 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1506.04924.

[19] V. Badrinarayanan, A. Kendall, and R. Cipolla, *SegNet: A deep convolutional encoder-decoder architecture for image segmentation*, 2016. arXiv: 1511.00561 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1511.00561.

[20] P. Tang, C. Zu, M. Hong, *et al.*, "DA-DSUNet: Dual attention-based dense su-net for automatic head-and-neck tumor segmentation in mri images", *Neurocomputing*, vol. 435, pp. 103–113, 2021, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2020.12.085. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231220319974.

[21] S. W. Chang and S. W. Liao, "KUnet: Microscopy image segmentation with deep UNet based convolutional networks", in *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019, pp. 3561–3566. DOI: 10.1109/SMC.2019.8914048.

[22] O. Ronneberger, P.Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation", in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, (available on arXiv:1505.04597 [cs.CV]), vol. 9351, Springer, 2015, pp. 234–241. [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a.

[23] N. Ketkar, J. Moolayil, N. Ketkar, and J. Moolayil, "Deep learning with python: Learn best practices of deep learning models with pytorch", *Apress LP*, 2020.

[24] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning", *CoRR*, vol. abs/2106.11342, 2021. arXiv: 2106.11342. [Online]. Available: https://arxiv.org/abs/2106.11342.

[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation", *CoRR*, vol. abs/1411.4038, 2014. arXiv: 1411.4038. [Online]. Available: http://arxiv.org/abs/1411.4038.

[26] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 2022. DOI: 10.1109/TNNLS.2021.3084827.

[27] T. M. Wani, T. S. Gunawan, S. A. A. Qadri, H. Mansor, F. Arifin, and Y. A. Ahmad, "Stride based convolutional neural network for speech emotion recognition", in *2021 IEEE 7th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)*, 2021, pp. 41–46. DOI: 10.1109/ICSIMA50015.2021.9526320.

[28] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai, "Gradient descent finds global minima of deep neural networks", in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 1675–1685. [Online]. Available: https://proceedings.mlr.press/v97/du19c.html.

[29] L. Bottou, "Stochastic gradient descent tricks", in *Neural Networks: Tricks of the Trade: Second Edition*, Springer, 2012, pp. 421–436.

[30] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson, "Mini-batch gradient descent: Faster convergence under data sparsity", in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 2880–2887. DOI: 10.1109/CDC.2017.8264077.

[31] P. Ruge, C. Birk, and B. Skrotzki, "Lineare algebra, nicht lineare gleichungen und interpolation", ger, in *HÜTTE Band 1: Mathematisch-naturwissenschaftliche und allgemeine Grundlagen für Ingenieure*, ser. Springer Reference Technik, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 143–171, ISBN: 9783662643686.

[32] J. Conklin, "Data science: A first introduction data science: A first introduction , by tiffany timbers, trevor campbell, and melissa lee. boca raton, fl: Crc press, 2022, xxiii + 420 pp., 133.84(*hardcover*),55.15 (paperback); isbn 978-0367532178: By tiffany timbers, trevor campbell, and melissa lee, boca raton, fl: Crc press, 2022, xxiii + 420 pp., 133.84(*hardcover*),55.15 (paperback); isbn 978-0367532178", eng, *Journal of Quality Technology*, vol. 55, no. 4, pp. 524–525, 2023, ISSN: 0022-4065.

[33] M. Azadkia, *Optimal choice of k for k-nearest neighbor regression*, 2020. arXiv: 1909.05495.

[34] *Metrics and scoring: Quantifying the quality of predictions*, Accessed: 17.06.2024. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html#r2-score.

[35] *R2*, Accessed: 17.06.2024. [Online]. Available: https://pages.stat.wisc.edu/~mchung/papers/R2.pdf.

[36] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: Current state and future challenges", *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011. DOI: 10.1109/MCOM.2011.6069707.

[37] A. Zhang, K. Zhu, R. Wang, and C. Yi, "Missing data inference for crowdsourced radio map construction: An adversarial auto-encoder method", in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, 2021, pp. 1–6. DOI: 10.1109/WCNC49053.2021.9417253.

[38] A. D. Kiureghian and O. Ditlevsen, "Aleatory or epistemic? does it matter?", *Structural Safety*, vol. 31, no. 2, pp. 105–112, 2009, Risk Acceptance and Risk Communication, ISSN: 0167-4730. DOI: https://doi.org/10.1016/j.strusafe.2008.06.020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167473008000556.