TECHNISCHE UNIVERSITÄT WIEN

institute of telecommunications

Dissertation

Wireless Localization via Learned Channel Features in Massive MIMO Systems

Author
Artan Salihu, M.Sc.
Matriculation Number: 11942320

Advisor
Assoc.Prof. Dipl.-Ing. Dr.techn. Stefan Schwarz

Assisting Advisor
Univ.Prof. Dipl.-Ing. Dr.techn. Markus Rupp

Examiners
Assistant Professor Aggelos Pikrakis, Eng, Ph.D.
Assoc.Prof. Dipl.-Ing. Dr. Klaus Witrisal

Institute of Telecommunications
Technische Universität Wien
Vienna, Austria

January, 2024

Abstract

Future wireless networks will evolve to integrate communication, localization, and sensing capabilities. This evolution is driven by emerging application platforms such as digital twins, on the one hand, and advancements in wireless technologies, on the other, characterized by increased bandwidths, more antennas, and enhanced computational power. Crucial to this development is the application of artificial intelligence (AI), which is set to harness the vast amounts of available data in the sixth-generation (6G) of mobile networks and beyond. Integrating AI and machine learning (ML) algorithms, in particular, with wireless localization offers substantial opportunities to refine communication systems, improve the ability of wireless networks to locate the users precisely, enable context-aware transmission, and utilize processing and energy resources more efficiently.

In this dissertation, advanced ML algorithms for enhanced wireless localization are proposed. Motivated by the capabilities of deep neural networks (DNNs) and the advancements in massive multiple-input-multiple-output (MIMO) systems, the dissertation aims to address some of the fundamental limitations of ML-based localization approaches related to dependability, generalization, and data scarcity. The dissertation has three main parts, each dedicated to addressing specific challenges and introducing new algorithms.

The first part of this dissertation focuses on improving the scalability and reliability of supervised learning techniques for wireless localization. Here, two variational DNN approaches are presented, designed to overcome the limitations in measuring the uncertainty of DNN-based position estimates in co-located massive MIMO systems. Further, this part extends the investigation to assess the localization accuracy in a distributed antenna system (DAS). It also introduces a strategy for selecting the most relevant subset of remote radio heads (RRHs) to alleviate the fronthaul overhead associated with dense wireless network deployments.

The second part of the dissertation is a transition from supervised to unsupervised learning. This part is concerned with subspace and metric-learning approaches that can learn low-dimensional channel features. Addressing the challenges related to data scarcity, this part introduces a contrastive task and a Siamese-based DNN to learn a four-dimensional channel representation that is useful for wireless localization. Compared to a base DNN classifier, the proposed method significantly improves localization performance, particularly in small data and non-line-of-sight (NLOS) conditions.

Finally, the third part of the dissertation reconsiders the foundational components and optimization strategies generally used in DNN-based localization methods. It first introduces a transformer-based model for more robust channel feature learning. It then builds upon the transformer-based method and proposes a non-contrastive self-supervised learning (SSL) approach. This part of the dissertation shows how to exploit the macroscopic and microscopic characteristics of the channel to achieve better transfer learning across various configuration settings, propagation environments, and wireless downstream tasks. Moreover, it investigates multiple variants of transformer-based models and uses multiple evaluation approaches and datasets to assess the localization accuracy in both co-located massive MIMO systems and a DAS.

Acknowledgments

Personal

I am profoundly indebted to Prof. Stefan Schwarz for his invaluable guidance and mentorship throughout my Ph.D. journey. His insights and expertise have significantly shaped my research approach and thinking. I sincerely thank Prof. Markus Rupp, whose constructive feedback and support, including securing the necessary funding, have been indispensable to my work.

I am deeply grateful to Prof. Aggelos Pikrakis and Prof. Klaus Witrisal for agreeing to review my dissertation and serving on the defense examination board.

I thank my colleagues at the Institute of Telecommunications for their stimulating discussions and shared moments of joy and challenge. Outside the institute, a special mention goes to Driton Statovci and Prof. Shkelzen Cakaj, who helped me to embark on this path.

I owe my thanks to my family. To my parents and my brother for their lifelong support and love. I am also thankful to my wife, Erza, whose patience and understanding know no bounds, and my daughter, Ana, the joy of my life, who brings laughter and happiness into each day.

Institutional

The Institute of Telecommunications has provided a vibrant academic environment that I have been fortunate to be a part of. My research would not have been possible without the generous financial support from the Christian Doppler Laboratory and its affiliated industrial partners, especially ÖBB and A1, for which I am immensely grateful. The financial support from the Austrian Federal Ministry for Labour and Economy and the National Foundation for Research, Technology and Development is gratefully acknowledged.

List of Abbreviations

1G first-generation

5G fifth-generation

6G sixth-generation

AE autoencoder

AI artificial intelligence

AoA angle of arrival

AUCO area under the confidence-oracle curve

BS base station

CAD computer aided design

CNN convolutional neural network

CSI channel state information

CTF channel transfer function

CU central unit

DAS distributed antenna system

DNN deep neural network

DR dimensionality reduction

ECDF empirical cumulative distribution

eMBB enhanced mobile broadband

GIS geographic information technology

GMM Gaussian mixture model

GP Gaussian process

GPS global positioning system

ISAC integrated sensing and communications

k-NN k-nearest neighbors

LBS location-based services

LOS line-of-sight

LUD Location and Uncertainty-Aware DNN

MC Monte Carlo

MDS multidimensional scaling

MIMO multiple-input multiple-output

ML machine learning

MLP multi-layer perceptron

mMTC massive machine-type communications

NLL negative log-likelihood

NLOS non-line-of-sight

NLP natural language processing

NN neural network

OFDM orthogonal frequency-division multiplexing

OSM OpenStreetMap

PCA principle component analysis

PHY physical layer

PPP Poisson point process

QoS quality of service

ROI region of interest

RP reference point

RRH remote radio head

RSD RRH subset selection DNN

RSS received signal strength

RT ray-tracing

SBR shooting and bouncing ray

SSL self-supervised learning

SWiT self-supervised wireless transformer

ToA time of arrival

UE user equipment

ULA uniform linear array

URLLC ultra-reliable low-latency communications

WiT wireless transformer

Contents

1

Introduction

Since the introduction of the first-generation (1G) of mobile networks, wireless communication has become an integral part of society, revolutionizing how we connect, access information, and engage in numerous aspects of our lives. The evolution from 1G to state-of-the-art fifth-generation (5G) technology has witnessed remarkable progress in services, transitioning from basic analog voice communication to high-speed data transmission, including enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable low-latency communications (URLLC). As indicated in the mobility report by Ericsson [1], global mobile subscriptions exceed 8.3 billion in 2023, with an expectation for total mobile data traffic to experience a threefold increase between 2022 and 2028. The report also anticipates that the 5G subscriptions will hit 4.6 billion by the end of 2027, making up more than 50% of all mobile subscriptions, thereby paving the way for a more interconnected world.

As we look beyond 5G and sixth-generation (6G), the envisioned future comprises the seamless integration of the physical and digital worlds [2, 3]. As a result, digital twins will emerge as fundamental platforms for novel applications, facilitating exhaustive digital representations of the physical and biological worlds across spatial and temporal dimensions. Realizing such a vision calls for further enhancements in wireless network capacity, a reduction in transmission latency, and integrating high-precision means of localization and sensing. Additionally, utilizing artificial intelligence (AI)-guided physical layer (PHY) optimization strategies will be crucial in materializing this vision. More specifically, the capabilities of mobile communications networks are anticipated to evolve far beyond their present role of connecting indi-

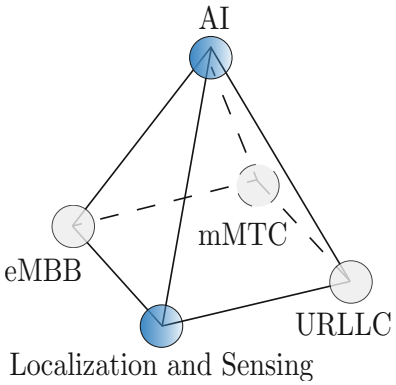


Figure 1.1: Location-aware and AI-enhanced wireless systems beyond 5G.

viduals or machines. In fact, they will encompass at least two additional categories of application scenarios [4], namely localization and sensing, and AI, as illustrated in Fig. 1.1. Wireless localization involves determining the position of an emitter based on processing the signals at one or more sensory-node receivers. Wireless sensing is a broader category of additional foreseen wireless system tasks, ranging from device-free target detection to environment reconstruction and imaging. On the other hand, AI, in more general terms, incorporates a range of techniques, primarily machine learning (ML) methods, which allow machines to *learn* from data and prior experiences. As such, AI-enhanced communication systems may allow adaptive data-driven decisions for various signal-processing parts in the communication chain.

Integrating AI with wireless localization presents an opportunity to optimize advanced communication systems to leverage the surroundings better. This integration can improve their ability to accurately locate the users, allow context-aware transmission, and more efficiently utilize processing and energy resources. In this context, the primary objective of this dissertation is to propose advanced ML algorithms for enhanced wireless localization.

1.1 Motivation and Scope of the Thesis

Wireless Localization

The capability to wirelessly pinpoint the location of a user equipment (UE) has grown from a niche service, mainly employed in military contexts, to a possible enabler supporting various wireless technologies and use cases. The most notable systems for wireless localization are *space-based*, specifically global navigation satellite systems (GNSS) [5,6]. Examples of GNSS include the global positioning system (GPS), which

is widely relied upon for various commercial and non-commercial applications [7]. Under favourable conditions, this system can yield localization accuracies of a few meters or even better. However, the quality of service (QoS) rendered by GPS is substantially compromised in environments that exhibit multipath effects and severe path loss due to signal blockages, such as those found in densely populated urban areas and indoors [8, 9]. As a result, efforts to enhance localization services have extended to *ground-based* wireless communication systems. Among the most prominent *ground-based* systems are those which have undergone advancements through the evolution of cellular standards [10]. Until early 2000s, research in cellular localization was largely driven by the need for improved emergency services (e.g., E911) [11, 12]. Since then, wireless localization methods made substantial progress, promising horizontal accuracy of less than 150 m, e.g., in fourth-generation long-term evolution systems (4G-LTE) [13, 14]. New approaches are proposed for different use cases, such as the internet of things (IoT) [15] and assisted driving [16, 17], while also serving in different deployment scenarios, such as indoor [18] and outdoor environments [19].

Despite the advancements in localization research and significant performance improvements over the decades, services with meter-level accuracy and very-high availability requirements are anticipated to be supported in 5G and beyond communication systems [20]. New use cases impose additional strict requirements regarding accuracy, availability, and coverage. For instance, augmented reality related services would require coverage of less than 10 meters, an accuracy of a few centimeters, and a latency below 20 ms. On the other hand, remote monitoring, asset tracking, and other IoT related services require less than 10 m accuracy and more than 1 km coverage [20]. Furthermore, most use cases require a guaranteed QoS with availability ranging from 80% to 99.9% [20]. Next-generation cellular systems are the most promising candidate to complement GPS to attain the QoS metrics of the applications and use cases of those mentioned above and similar. Therefore, enhancing wireless localization performance is a fundamental and continuous research challenge.

The sensing capability of the network is increasingly recognized as the key driver for the 6G and the development of smart environments [4]. Such functionality is anticipated to be useful for a variety of location-aware applications, e.g., autonomous vehicles [21, 22]. Furthermore, with the research trend towards integrated sensing and communications (ISAC) [23], networks will leverage location information for self-optimization to improve efficiency and sustainability. ISAC aims to design joint systems capable of simultaneously performing communication and remote sensing. To achieve this, signal processing algorithms should ideally utilize the same signal or waveform for both sensing and communication tasks. Consequently, it becomes crucial to develop algorithms that can improve wireless localization and use the same signal information as for other wireless communication tasks.

Data-Driven Localization

Localization methods can be divided into two categories: model-based and data-driven [24]. Model-based techniques require knowledge of the geometric relationship between the estimated parameters of the received signal and the position of the transmitter [25]. Therefore, the performance of the existing model-based methods is heavily degraded when such a relationship is not available, e.g., in non-line-of-sight (NLOS) or dense multi-path propagation conditions [25–28]. Consequently, data-driven approaches, specifically ML, have emerged over the years [29]. Among the various ML approaches, deep neural network (DNN)-based models have demonstrated exceptional localization accuracy [30–35].

Traditionally, wireless localization methods use derived features of the estimated channel at the base station (BS), such as signal time of arrival (ToA), angle of arrival (AoA), received signal strength (RSS), or a combination of them [27, 36, 37]. Conversely, DNN-based methods prefer utilizing the channel state information (CSI) in its acquired form [30,31,34,38,39]. The acquisition of CSI is particularly important for advanced communication systems that use massive multiple-input multiple-output (MIMO) [40, 41]. Hence, CSI is, in general, readily-available for localization. Massive MIMO, characterized by a large antenna array at the BS, is widely regarded as the key technology for 5G [42, 43] and forthcoming communication systems [44]. Employing a considerable number of antennas enhances the angular resolution of the received multipath signal, thereby benefiting localization methods [39, 45–50].

Massive MIMO systems typically adopt a centralized implementation approach, wherein the antennas are geometrically co-located at the BS. Another implementation strategy that has gained research attention involves the utilization of spatially distributed antennas, notably the distributed antenna system (DAS) [51, 52]. In this case, a large number of geographically spread out antennas are used by means of remote radio heads (RRHs) in order to extend the BS antenna ports [52]. Due to the inherent spatial diversity, DAS can also be favorable for the ML-based wireless positioning techniques [53, 54].

ML-based methods can generally be supervised or unsupervised, depending on whether labelled training data is necessary. The vast majority of DNN localization techniques are supervised and constrained to a task-specific feature learning (see Sec. 1.2 for relevant literature), raising concerns, particularly about the ability to work across diverse scenarios, and adapt in data-scarce environments. Hence, developing DNN methods that sustain good accuracy and transferability remains an essential research topic.

Scope of Work

Motivated by the capabilities of DNNs and the advancements in massive MIMO systems, we seek to address some of the fundamental limitations of ML-based localization approaches related to dependability, generalization, and data scarcity. We propose DNN models that can learn channel parameters useful to infer various quantities of a wireless network task, particularly: the UE location. More concretely, the dissertation has three main parts organized into three different chapters, each dedicated to addressing specific challenges and introducing new algorithms:

- In Chapter 3, we deal with enhancing supervised techniques, such as overcoming challenges related to dependability and scalability. We consider the uplink channel estimates at a co-located massive MIMO system and a DAS. We introduce two variational methods to quantify the uncertainty of DNN-based position estimates, along with a strategy to determine the most relevant subset of RRHs for localization.
- In Chapter 4, we transition from supervised learning to approaches that seek to learn the intrinsic properties of the high dimensional channel estimated at a massive MIMO BS, especially in situations with limited labelled data. Addressing the challenges related to data scarcity, we propose a metric-learning technique to learn channel features, and we also study the effect of NLOS conditions.
- In Chapter 5, we emphasize the challenges of generalization and transferability in current supervised and unsupervised approaches. Rethinking the foundational components and optimization strategies generally used in DNN-based localization methods, we introduce an approach for self-supervised learning of channel representations. Here, we aim to exploit the macroscopic and microscopic characteristics of the channel to achieve better transfer learning across various configuration settings and propagation environments.

1.2 Literature Review

1.2.1 Fully Supervised Location Estimates

Numerous works propose supervised training methods, where the utilized channel features are labelled with location information (e.g., position coordinates of the target) [31, 34, 38, 39, 49, 55]. These techniques establish a mapping function between the obtained CSI and the corresponding position coordinates intending to achieve accurate localization performance on new, unseen data. However, DNN methods for localization output blind estimates while failing to give any useful information about

their predictive uncertainty. This differs from well-known probabilistic ML approaches, e.g., Gaussian process (GP)-based methods, which inherently provide uncertainty estimates as shown in [54,56,57]. Overconfident incorrect predictions in safety-critical applications can have tragic consequences; hence, capturing and reasoning about the uncertainty of estimated positions is fundamental to integrating DNN methods in wireless localization systems. We discuss and address the dependability of DNN-based location estimates in Chapter 3.

The majority of data-driven localization methods are proposed for massive MIMO in a co-located setup. However, the investigation regarding the positioning abilities of DNN-based methods in a DAS remains limited, with notable exceptions such as the study in [34]. The works in [53, 54] make use of RSS and ML based on GPs, while [58, 59] investigate other conventional out-of-the-box ML techniques and exploit different channel information. Previous works evaluate the signal information at a central unit (CU) from all connected RRHs. However, this approach can lead to increased fronthaul overhead and computational complexity at the CU, which presents a scalability challenge in dense wireless network deployments. In Chapter 3, we present a deep learning method with the capability of RRH subset selection, UE localization, and uncertainty estimation.

1.2.2 Deep Metric Learning

In general, supervised DNN-based methods excel in any task where extensive labelled datasets are readily available for training. However, collecting large-scale geo-tagged channel estimates for different mobile network tasks can be time-consuming, error-prone, and in many cases, impractical. Consequently, several studies have relied on conventional dimensionality reduction (DR) techniques [60], and autoencoders (AEs) [61] for wireless localization [30,62–66]. Such methods essentially attempt to compute a lower-dimensional channel space in an unsupervised manner, while retaining non-redundant features and structures or suppress the noise from high-dimensional CSI acquired at the BS.

Assuming a subset of CSI has labels, its low-dimensional features can either serve as a reference map [64–66] or input to a task-specific model [30] to derive the final location of the UEs. The authors in [64–66] aim to capture the CSI manifold and project it onto, for instance, a two-dimensional channel map (recently known as channel charting methods [65]). The reference map, in some cases, preserves the local structure or relative distances between the UEs from which the channel is evaluated. Provided such maps only yield pseudo-locations, a matching algorithm like k-nearest neighbors (k-NN) can be utilized to compare a new transmitter's channel features to the known locations in the channel chart and approximate its position.

AEs employ neural networks, as hierarchical feature extractors, to obtain the latent space representation. In general, this is achieved using a reconstruction objective [61].

Conversely, other DR techniques [60], including manifold learning methods like multidimensional scaling (MDS), strive to preserve specific metrics, e.g., the pairwise distances between the CSI obtained at different locations, which can reveal the inherent geometric relationships in the original data space. A more contemporary approach, deep metric learning, can combine the feature learning capabilities of AEs with the metric-preservation flexibility of other manifold-learning techniques. Methods based on deep metric learning have shown improved performance with limited training samples in object detection [67], object similarity [68], and clustering [69]. Approaches motivated by the so-called Siamese networks [70] have been proposed for wireless channel charting [71]. In Chapter 4, we present a deep metric learning approach, and assess the impact of NLOS conditions and the training sample size.

1.2.3 Transformer and Self-Supervised Features

The learning capability of DNN-based models depends not only on the optimization objective but also on the configuration of layers and the organization of such layers within the neural network architecture. Convolutional neural networks (CNNs) and multi-layer perceptrons (MLPs) have been the main components of many DNN models for decades. CNNs are particularly well-suited at processing grid-like organized signals (e.g., images) [72]. On the other hand, MLPs, consisting of layers of units in a fully-connected, directed graph-like approach, have been foundational blocks of DNN models since the 1980s [73].

As the system bandwidth and the number of antenna elements at the BS becomes larger [74], the dimensionality of CSI also increases. This can pose a challenge for methods that rely solely on MLPs, such as insufficient number of units in the input layer. Increasing the units would increase the number of parameters. As a consequence, the capacity of the model would become higher, and therefore a more extensive training set is needed. Furthermore, MLPs cannot capture local correlations [75], e.g., the channel at neighboring antennas or subcarriers, information that would be common even for hand-feature extractors. Finally, due to their fully-connected nature, they have no mechanism to ensure invariance with respect to small-scale variations in the input channel. Hence, they might not be the optimal choice for channel representation learning or channel-to-location mapping.

Numerous works suggest utilizing CNNs to better capture the channel characteristics relevant for the channel-to-location mapping [31, 35, 50]. However, while CNNs are primarily designed and appropriate for structured signals, they introduce a strong inductive bias by utilizing fix-sized filters to *slice* the channel and learn different portions of the input channel representation.

To cope with the issues of imperfect channel estimates, and other system impairments, various works suggest the conventional option of hand designing feature extractors for

more robust models. A common idea in the literature for hand-engineered features is to leverage the channel transform domains, such as angle, delay or Doppler, e.g., [49, 76]. However, hand-crafting input features limit the expressive capacity of DNN models, hence constraining the generalization of learned representations or the trained model.

In contrast to long-standing approaches, transformer-based architectures proposed more recently in natural language processing (NLP) [77] and computer vision [78] adopt the *attention* mechanism [79]. Consequently, they show greater learning capacity [80], less inductive bias, and can capture local and wide-range dependencies. Historically, the *attention* mechanism was used on top of convolutional feature maps, which is reflected in recent growing literature in wireless localization [81], or even a combination with long short-term memory (LSTM) [82]. In Chapter 5, we propose a transformer-based model and then use it to design a new channel representation learning method based on a self-supervised paradigm. In contrast to the deep metric learning, we elaborated in Sec. 1.2.2, the self-supervised learning leverages unlabelled data as a form of supervision and relies only on multiple views of the same input signal information. The latter has multiple benefits concerning computational cost and, more importantly, generalization and transferability.

1.3 Structure and Contributions

In this section, we outline the structure of the dissertation and its main contributions.

Chapter 2 – System Model and Problem Formulation

In the first chapter, we present an overview of the system model, which is the basis for the contributions detailed in the following chapters. We detail the signal model for an uplink wireless transmission system. The localization methods we elaborate on throughout the dissertation consider a multicarrier system and assume that the receiver, the BS, has many antennas. Finally, we formalize the metrics used to evaluate the wireless localization performance of the proposed methods.

Chapter 3 – Towards Dependable Location Estimates

We begin the second chapter by addressing the challenges in existing DNN wireless localization methods that depend on fully-supervised learning. First, we introduce uncertainty-aware and scalable DNN approaches to measure the uncertainty of their location estimates when propagation conditions change and a finite number of training samples is available at a co-located MIMO system. Then, we shift to investigating the localization accuracy in a DAS. Finally, we extend the uncertainty-aware method

to incorporate a learning strategy that determines the most relevant subset of RRHs for localization. The contributions of this chapter are:

- Introduce two variational and scalable DNN approaches for quantifying location estimate uncertainties.
- Evaluate the performance on publicly available ray-tracing datasets for indoor and outdoor environments.
- Show that data uncertainty captures NLOS effects, while model uncertainty enhances the overall reliability.
- Propose a learning-based RRH selection algorithm for end-to-end training with a discrete set of RRHs.
- Demonstrate that the selection strategy is effective in strong multipath environments.

The contributions of this chapter have been published in [83, 84]:

[i] A. Salihu, S. Schwarz, and M. Rupp, "Towards scalable uncertainty aware DNN-based wireless localization," in 2021 29th European Signal Processing Conference (EUSIPCO), 2021, pp. 1706–1710.

[ii] A. Salihu, S. Schwarz, and M. Rupp, "Learning-based remote radio head selection and localization in the distributed antenna system," in 2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit). IEEE, 2022, pp. 65–70.

Chapter 4 – From Shallow to Deep Metric Learning for Localization

In this chapter, we shift the focus towards the dimensionality reduction of CSI and unsupervised learning. Here, we aim to obtain a low-dimensional channel representation, such that transmitters in close proximity are mapped to nearby points on a manifold. We present a contrastive deep metric learning method that aims to learn channel features based on a *distance* metric between two UE locations. The presented method encourages CSI embeddings of neighboring UEs to be close and push apart those of non-adjacent ones in terms of a pre-defined distance metric. The contributions of this chapter are:

- Investigate a classical DR approach for determining the location of a line-of-sight (LOS) UE from its CSI sensed at a large-antenna array BS.
- Introduce a deep metric-learning method based on a contrastive objective function for low-dimensional channel representation.

- Reveal that the absence of a LOS path has little impact on the localization performance based on simulations.
- Show that the introduced method requires fewer training samples when compared to a fully-supervised DNN method.

The contributions of this chapter have been published in [85, 86]:

[i] A. Salihu, S. Schwarz, and M. Rupp,"Semi-supervised localization utilizing CSI at large antenna array base stations," in WSA 2020; 24th International ITG Workshop on Smart Antennas, 2020, pp. 1–5.

[ii] A. Salihu, S. Schwarz, A. Pikrakis, and M. Rupp, "Low-dimensional representation learning for wireless CSI-based localization," in 16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). IEEE, 2020, pp. 1–6.

Chapter 5 – Wireless Transformer and Self-Supervised Representations

In the fifth chapter, we address the limitations of previous work in previous chapters and aim to provide some foundation work for future wireless channel representation learning. First, we introduce a transformer-based model that maintains and exploits the per-subcarrier channel structure in a MIMO-OFDM system. Then, we propose a non-contrastive self-supervised learning method. The proposed method ingests channel realizations and aims to transform them into representations that are invariant to fading and system impairments, such that they may be exploited for positioning the UE and can achieve transfer learning to different scenarios. The contributions of this chapter include:

- Propose a transformer model and investigate its localization ability in a fully-supervised setting.
- Introduce a self-supervised *joint embedding* model to forsake the dependency on contrastive examples for wireless channel representation learning.
- Incorporate a learning module to exploit microscopic and macroscopic channel fading characteristics.
- Investigate the impact of channel transformations on the quality of learned embeddings.
- Show that the self-supervised model can outperform fully-supervised techniques in small data regimes, sometimes even with a linear model with negligible computational complexity.

- Study the transferability of the method to new environments and other wireless downstream tasks.

The contributions of this chapter have been published in [87, 88]:

[i] A. Salihu, S. Schwarz, and M. Rupp, "Attention aided CSI wireless localization," in 2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC), 2022, pp. 1–5.

[ii] A. Salihu, M. Rupp, and S. Schwarz, "Self-supervised and invariant representations for wireless localization," IEEE Transactions on Wireless Communications, pp. 1–1, 2024.

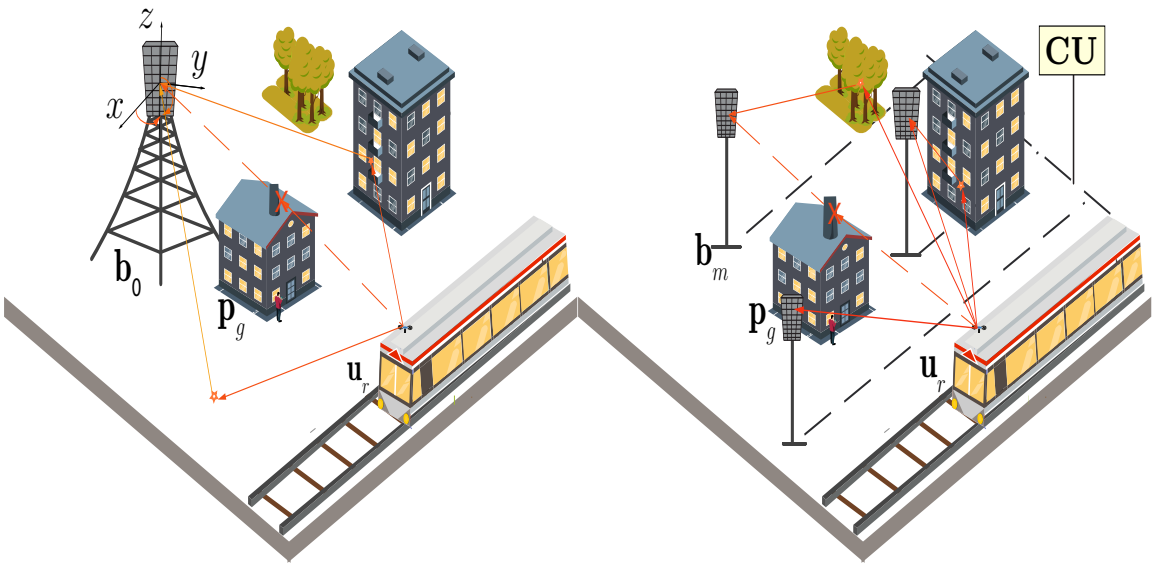
2

System Model

In this chapter, we describe a massive MIMO system model utilized throughout this dissertation. First, we illustrate the input-output relationship of the signal for an uplink transmission involving a single UE and a BS. Next, we elaborate on the wireless channel model and detail modeling of the uncertainty of the channel parameters. As we navigate through the chapters, it is relevant to note that the formulation of the signal model varies depending on the aspect being investigated. Finally, in this chapter, we define the performance evaluation metrics. The system model described in this chapter was used in the works published in [87, 88].

2.1 System Model

Throughout the dissertation, we consider a massive MIMO uplink system. In Fig. 2.1, we illustrate two scenarios: one showing a centralized antenna setup and the other demonstrating a distributed antenna configuration. Hence, we either consider a co-located massive MIMO or a DAS network. For the first one, we consider that a BS located at $\mathbf{b}_0 = [b_{0,1}, b_{0,2}, b_{0,3}]^T$ is equipped with an array of N_r antenna elements. For the second case, i.e., DAS, M spread out RRHs (or access points) serve the UEs in the area. Similarly, RRH $m \in \{1, \ldots, M\}$, positioned at $\mathbf{b}_m = [b_{m,1}, b_{m,2}, b_{m,3}]^T$, has an array of N_m antenna elements, and the total number of antennas in a DAS is $N_r = \sum_{m=1}^{M} N_m$. Such a network setup, characterized primarily by the absence of traditional cell boundaries, resembles what is commonly referred to in the literature as 'cell-free massive MIMO' or 'network MIMO' [89]. More specifically, we consider that



(a) Co-located antennas.

(b) Distributed antennas.

Figure 2.1: Illustration of the system model as considered in this dissertation when the scenario employs a) a co-located massive MIMO system or b) a DAS.

the signal between an UE and all active RRHs is available at a CU for joint signal processing and UE location estimation. We assume that all RRHs are connected via high-speed fronthaul links to the CU, i.e., synchronization delay between the RRHs and the CU is considered negligible. Furthermore, we assume that the active UE, denoted by $r \in \{1, \ldots, R\}$, has a single-antenna placed in three-dimensional space, i.e., $\mathbf{u}_r = [u_{r,1}, u_{r,2}, u_{r,3}]^T$ denotes the position vector. Lastly, we assume that within the region of interest (ROI), G objects are located at positions $\mathbf{p}_g = [p_{g,1}, p_{g,2}, p_{g,3}]^\top$, for $g \in \mathcal{G}$ and $\mathcal{G} = \{1, \ldots, G\}$. These objects may obstruct, reflect, or scatter the propagated signals between the transmitter and receiver.

2.1.1 Signal Model

We consider orthogonal frequency-division multiplexing (OFDM) transmissions such that the channel transfer function (CTF) used for localization is sampled over time (OFDM symbols) and frequency (OFDM subcarriers). Let N_c be the total number of subcarriers and the sampling interval $T_{\text{samp}} = \frac{1}{B_{\text{sys}}}$, where B_{sys} is the system (channel) bandwidth. We assume that any $N_c' < N_c$ subcarriers can be used for synchronization and channel estimation, e.g., pilot subarriers, and may also be utilized for UE position estimation. We consider that in practice the maximum channel delay $\tau_{\max}$ is less than the symbol duration, $\tau_{\max} < T_s$. Hence, for the received signal at the BS, denoted

as $\mathbf{y}_n \in \mathbb{C}^{N_r \times 1}$, corresponding to the subcarrier $n \in \{1, \ldots, N_c\}$, we can write the input-output relationship as

$$\mathbf{y}_n = \sqrt{P_{\text{tx}}}\tilde{\mathbf{h}}_n x_n + \mathbf{n}_n \,. \tag{2.1}$$

Here, P_{tx} is the average transmit power, x_n is the normalized transmitted signal with $|x_n|^2 = 1$, $\tilde{\mathbf{h}}_n \in \mathbb{C}^{N_r \times 1}$ is the channel on subcarrier n, and $\mathbf{n}_n \sim \mathcal{CN}(\mathbf{0}, N_0 W_{\text{sc}} \mathbf{I}_{N_r})$, where N_0 is the noise power spectral density and W_{sc} is the subcarrier bandwidth. We assume that the BS is able to estimate the channel vectors $\{\tilde{\mathbf{h}}_n\}_{\forall n}$ through uplink pilot signals, and can mitigate pilot contamination [90, 91]. Finally, we denote the estimated CSI as $\mathbf{h}_n$, including considerations of channel estimation error, which we discuss in Sec. 2.1.3.

2.1.2 Channel Model

Although the DNN methods proposed and elaborated throughout this dissertation are data-driven and not tied to a specific channel model, we study the impact of various propagation factors on the performance of the algorithms on location estimation. Consequently, it is beneficial to characterize the channel using location-related parameters like distances and angles for investigation. This is achieved by employing a geometric channel model [92–94]. More specifically, we use a discrete-path model commonly applied in ray-tracing tools for channel synthesis, which involves representing the channel as a superposition of a finite number of propagation paths between the transmitter and receiver. The propagation path parameters, in this case, refer to the mean values of the parameters of multiple rays that are combined in a propagation path $\ell \in \{1, \ldots, L_{\text{path}}\}$. Hence, as a general relationship between the user location and the estimated parameters of the channel for such scenarios, as illustrated in Fig. 2.1, we consider

$$\tilde{\mathbf{h}}_n = \sum_{\ell=1}^{L_{\text{path}}} \eta_\ell e^{-j2\pi n \Delta f \tau_\ell} \mathbf{\Gamma}\left(\varphi_{az,\ell}, \varphi_{el,\ell}\right) \,. \tag{2.2}$$

In (2.2), η_ℓ and τ_ℓ denote the $\ell-$th path's complex gain and propagation delay between the $r-$th UE location and the BS (or $m-$th RRH). The AoAs in azimuth and elevation are denoted by $\varphi_{az,\ell}$ and $\varphi_{el,\ell}$, respectively. Although it is challenging to argue the resolvability of multiple paths at lower frequencies, and hence their relationship to the geometry of the environment, in mmWave and beyond, there is a stronger relationship between the individual received paths and the geometric information of the channel [28, 95]. Finally, assuming a uniform planar array at the BS with N_{r_x} and N_{r_z} antenna elements along the $x-$ and $z-$axis, the expression for the array response vector for the elaborated example at the receiver can be written

as

$$\mathbf{\Gamma}\left(\varphi_{\mathrm{az}}, \varphi_{\mathrm{el}}\right) = \mathbf{\Gamma}_z\left(\varphi_{\mathrm{el}}\right) \otimes \mathbf{\Gamma}_x\left(\varphi_{\mathrm{az}}, \varphi_{\mathrm{el}}\right) , \tag{2.3}$$

The array steering vectors $\mathbf{\Gamma}_x(\cdot)$, and $\mathbf{\Gamma}_z(\cdot)$ are

$$\mathbf{\Gamma}_x\left(\varphi_{\mathrm{az}}, \varphi_{\mathrm{el}}\right) = \left[1, e^{j\frac{2\pi}{\lambda_c} d_{\mathrm{ant}} \sin(\varphi_{\mathrm{el}}) \sin(\varphi_{\mathrm{az}})}, \ldots, e^{j\frac{2\pi}{\lambda_c} d_{\mathrm{ant}} (N_{r_x}-1) \sin(\varphi_{\mathrm{el}}) \sin(\varphi_{\mathrm{az}})}\right]^T , \tag{2.4}$$

$$\mathbf{\Gamma}_z\left(\varphi_{\mathrm{el}}\right) = \left[1, e^{j\frac{2\pi}{\lambda_c} d_{\mathrm{ant}} \cos(\varphi_{\mathrm{el}})}, \ldots, e^{j\frac{2\pi}{\lambda_c} d_{\mathrm{ant}} (N_{r_z}-1) \cos(\varphi_{\mathrm{el}})}\right]^T ,$$

with $\lambda_c = c/f_c$, where f_c being the carrier frequency and c the speed of light, and $d_{\mathrm{ant}} = \lambda_c/2$ the antenna element spacing.

2.1.3 Uncertainty Model

In Chapter 5, when we generate synthetic channel data, we consider scenarios in which a subset of scattering objects vary its position over the time $t \in \{1, \ldots, T\}$ (e.g., moving objects in the environment). Specifically, by fixing the positions of the receiver and transmitter, we realize time-varying conditions of the environment by altering the positions of G' scattering objects, where $G' = |\mathcal{G}'|$ and $\mathcal{G}' \subseteq \mathcal{G}$. Thus, we have

$$p_{g,i}^{(t)} \triangleq p_{g,i} + \mathring{w}_{g,i}^{(t)}, \tag{2.5}$$

where $\mathring{w}_{g,i}$ is a zero-mean Gaussian noise with variance $\sigma_{\mathring{w}}^2$ at $i-$th coordinate. Moreover, we also introduce objects randomly into the environment, simulating temporary obstacles, such as a train passing by. Doing so simulates a potential sudden block of the LOS or other dominant propagation paths. Similarly, we account for the variations in the position of antennas of the transmitter,

$$u_{r,i}^{(t)} \triangleq u_{r,i} + \dot{w}_{r,i}^{(t)}, \tag{2.6}$$

where $\dot{w}_{r,i}$ denotes a zero-mean Gaussian noise with variance $\sigma_{\dot{w}}^2$ at $i-$th coordinate. Note that the variations in the position of the scatterers alter the gain, delay and angle information of the individual multi-bounce NLOS paths, allowing for modelling the individual path uncertainties that lead to a non-additive distortion model [44]. Moreover, the uncertainty in the position of the antenna allows us to account for the effect of imperfect channel estimates due to, e.g., lack of perfect synchronization.

Further, we consider that the electromagnetic properties of the scattering objects change over time, which impacts the amplitude gain of the radar cross section (RCS) of the scattering objects. We assume that material types can randomly change and have a permittivity value of $\epsilon_{\kappa_{\mathrm{per}}}$ at time t, where κ_{per} is the type of the material [96]. Finally, we also consider atmospheric attenuation in the environment. Therefore, in

case of a rainy period $\boldsymbol{\mathcal{R}}$, which occurs with probability $\mathbb{P}(\boldsymbol{\mathcal{R}})$, we assume additional attenuation to the line-of-sight (LOS) path [97].

Finally, we denote by $\mathbf{H}_r^{(t)}$ a collection of N_c' subcarriers for $r-$th user location during a single-time snapshot, which we write for convenience in a matrix form as

$$\mathbf{H}_r^{(t)} = \begin{bmatrix} h_{1,1}^{(t)} & \dots & h_{1,N_c'}^{(t)} \\ \vdots & \ddots & \vdots \\ h_{1,N_r}^{(t)} & \dots & h_{N_r,N_c'}^{(t)} \end{bmatrix} \in \mathbb{C}^{N_r \times N_c'}. \tag{2.7}$$

We consistently omit the index t from mathematical expressions as we predominantly consider a single-realization of the channel (a snapshot) in our algorithms. Similarly, the index r may also be dropped when it does not particularly influence the discussion or analysis at hand. Finally, the input to a neural network-based models that we propose in this dissertation is real-valued; hence we handle complex-valued channel coefficients by stacking their real and imaginary parts.

2.2 Problem Formulation

In this dissertation, we propose models based on artificial neural networks (NNs). NNs are computational models largely inspired by the human brain [98]. Another way of thinking about artificial neural networks is to view them as a set of functions organized into a hierarchy, often referred to as *layers*. Each layer consists of numerous basic computational units that are called *neurons* [99]. By processing inputs through these layers, NNs can be trained to learn and recognize non-trivial patterns within data. Hence, for a given a training dataset composed of $\{\mathbf{H}_r, \mathbf{u}_r\}_{r=1}^{R}$, we can formulate training of neural network in a supervised settings as

$$\boldsymbol{\Theta}^{\star} = \arg\min_{\boldsymbol{\Theta}} J(\boldsymbol{\Theta}), \tag{2.8}$$

where the cost function

$$J(\boldsymbol{\Theta}) := \sum_{r=1}^{R} \mathcal{L}\left(\mathbf{u}_r, f_{\boldsymbol{\Theta}}(\mathbf{H}_r)\right). \tag{2.9}$$

The function $\mathcal{L}(\cdot)$ denotes a loss function, and $f_{\boldsymbol{\Theta}}(\mathbf{H}_r)$ is the mapping function imposed by a neural network parameterized by $\boldsymbol{\Theta}$. Then, we write the estimated UE location as $\widehat{\mathbf{u}}_r = f_{\boldsymbol{\Theta}}(\mathbf{H}_r)$.

2.2.1 Performance Metrics

We assess the performance of the wireless localization task primarily using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). MAE is calculated as

$$\text{MAE} = \frac{1}{R_{\text{test}}} \sum_{r'=1}^{R_{\text{test}}} \|\hat{\mathbf{u}}_{r'} - \mathbf{u}_{r'}\| \ , \tag{2.10}$$

where R_{test} are the number of locations to be estimated, i.e., the test UE locations. Similarly, for RMSE, we evaluate

$$\text{RMSE} = \sqrt{\frac{1}{R_{\text{test}}} \sum_{r'=1}^{R_{\text{test}}} \|\hat{\mathbf{u}}_{r'} - \mathbf{u}_{r'}\|^2} \, . \tag{2.11}$$

We also use empirical cumulative distribution (ECDF) and report the 95-th percentile of the error distribution.

3

Towards Dependable Location Estimates

The design and operation of ML algorithms rely significantly on the training dataset, which, in turn, is influenced by site-specific factors, including antenna distribution, hardware imperfections, and the granularity of surveyed locations. Additionally, the propagation channel can be nonstationary and often unpredictable. Analyzing and considering all possible situations during training a DNN model presents an intractable task. Hence, their dependability and scalability remain a key concern despite deep learning-based localization methods demonstrating improved positioning performance.

In this chapter, we deal with the challenges of uncertainty estimation of supervised DNN-based localization methods as well as the scalability issues arising from ultra-dense deployments of RRHs. The first part of the chapter introduces and evaluates two scalable, uncertainty-aware DNN-based localization methods. The methods are designed to measure uncertainty stemming from varying propagation conditions and the finite number of training samples. Subsequently, the second part of the chapter extends the functionality of the methods by integrating a *neural* combinatorial module. This module aims to select a subset of RRHs deemed most relevant for wireless localization.

3.1 Base DNN and Uncertainty Information

We begin by considering a straightforward and relatively low-complexity feedforward neural network, i.e., an MLP shown in Fig. 3.1. Throughout the dissertation, we will refer to it as the base DNN, as we will use it repeatedly to compare it with the proposed approaches in this dissertation or as a basis for newly proposed models. In the following, we set the number of hidden layers to $L = 4$ and the number of units for each hidden layer to 650. Selecting the number of hidden layers and units is more of an experimental decision than one grounded in theory. We make adjustments often based on empirical performance and the computational considerations. The standard components (i.e., the input layer and the activation functions) of the fully connected feedforward MLP require flattened, real-valued inputs, and real-valued. Hence, in this chapter, we treat the complex-valued channel as two independent real numbers, and utilize a single subcarrier to maintain a lower complexity of the DNN. Therefore, the channel for the input layer is $\bar{\mathbf{h}}_r = \mathbf{h}_n \in \mathbb{R}^{2N_r}$, and $n = 1$. Selecting a single subcarrier is typically sufficient representation in presence of strong LOS path and large coherence bandwidth. In a more pronounced selective fading channel, averaging over N_c subcarriers can potentially smooth out the effects of deep fading. However, both approaches inevitably result in information loss. We address this in Chapter 5 where we consider the whole estimated channel for the models presented.

Next, we define the localization problem as a regression task and the DNN as a function $f^{(\text{Base})}_{\mathbf{\Psi}_b} : \mathbb{R}^{2N_r} \mapsto \mathbb{R}^D$ parameterized by $\mathbf{\Psi}_b$, where we consider $D = 2$. Given the input of the DNN is the channel state vector $\bar{\mathbf{h}}_r$, we aim to directly map it to position information, $\mathbf{u}_r \in \mathbb{R}^D$. For a training dataset of R sample pairs, $\mathcal{D} = \{\bar{\mathbf{h}}_r, \mathbf{u}_r\}_{r=1}^{R}$, the set of optimal parameter values $\mathbf{\Psi}_b$ is learned by minimizing a given loss function, $\mathcal{L}_b(\cdot)$. Usually, the supervised training for the regression is performed to minimize the sum of squared errors,

$$\mathcal{L}_b(\mathbf{u}_r, \bar{\mathbf{h}}_r, \mathbf{\Psi}_b) = \mathbb{E}_{(\bar{\mathbf{h}}_r, \mathbf{u}_r)\sim\mathcal{D}} \left[\left\| \mathbf{u}_r - f^{(\text{Base})}_{\mathbf{\Psi}_b}(\bar{\mathbf{h}}_r) \right\|^2 \right], \tag{3.1}$$

where $\mathbb{E}_{(\bar{\mathbf{h}}_r, \mathbf{u}_r)\sim\mathcal{D}}$ is the expected value or average computed over a sampled batch of training data. Such a DNN-based approach yields fully deterministic neural network parameters that output only point estimates of the network, i.e., $\hat{\mathbf{u}}_r = f^{(\text{Base})}_{\mathbf{\Psi}_b}\left(\bar{\mathbf{h}}_r\right)$. One can interpret this as outputting the mean of a probability distribution while disregarding other moments. However, providing precise position estimates for the UE while including uncertainty information is critical to evaluate the confidence of the model. Incorporating this information facilitates continuous learning by identifying and focusing on measurements with higher uncertainty, e.g., importance sampling. These are highly desired features for localization approaches applied to real-world and safety-related tasks in railroad transportation, vehicular communications, and assets tracking, to name a few. Conventional Bayesian neural networks (BNNs) can provide estimates of the posterior variance on the network weights in a principled manner [100].

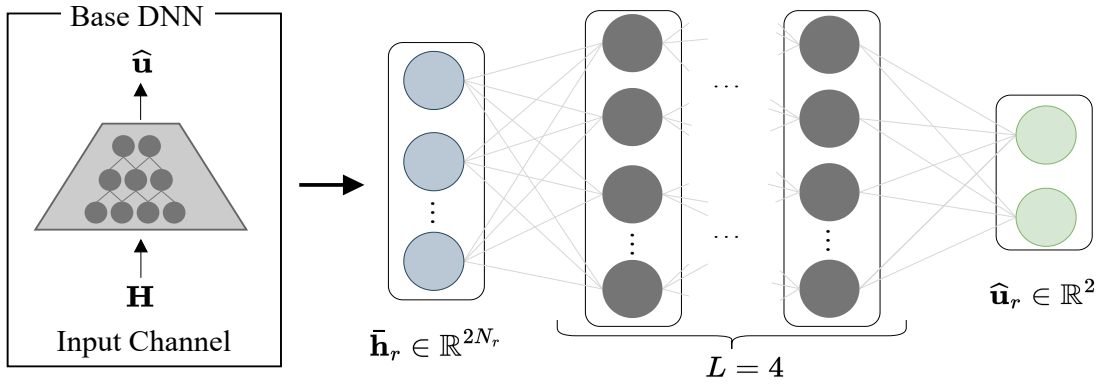


Figure 3.1: A basic MLP, termed as the base DNN in this dissertation. It serves as a *backbone* for the methods in this chapter and occasionally as a reference for comparison throughout the dissertation.

However, they substantially increase the number of parameters, require different optimization strategies, and become hard to train for large datasets [101]. More recently, Monte Carlo (MC) sampling techniques have emerged as an alternative to estimate the uncertainty in the model parameters that can scale for modern DNNs and large-scale datasets [102–104]. In the following, we aim to formulate a scalable and variational approach to accurately locate the UE and provide confidence information while maintaining a low computational complexity.

3.2 LUD: Location and Uncertainty-Aware DNN

Uncertainty predominantly stems from two sources, i.e., data and model uncertainty. The data uncertainty (also known as *aleatoric* uncertainty) is mainly related to the stochastic nature of the wireless signal (e.g., noise and interference) but also to the propagation environment, such as the availability of LOS and multipath fading. The latter, the model uncertainty (also known as *epistemic* uncertainty), results from unbalances in the training data distribution, e.g., the unpredictable changes in channel conditions. For instance, a model trained on samples with infrequent cases of specific channel conditions should exhibit increased model uncertainty compared to samples that often occurs in the training dataset. Furthermore, regions surveyed less or not at all during the data collection phase (e.g., an inaccessible room in a train station) should output higher uncertainty values compared to regions with abundant training data. Therefore, both types of uncertainty are relevant for wireless localization. Indeed, we cannot assume the presence of LOS, ensure that the training dataset covers all possible areas, or anticipate the ambiguous paths the signal may travel. Hence, we model the DNN to explicitly learn the underlying uncertainty from the input data. We first acquire data and model uncertainty separately. Finally, we

also combine both uncertainties to acquire a total confidence score into one end-to-end model. The model is coined as LUD, i.e., Location and Uncertainty-Aware DNN.

3.2.1 Data Uncertainty

The data uncertainty is a property of the data itself, and we can train the network to directly output the parameters of a probability distribution. To do so, we use a Gaussian mixture model (GMM). In this case, LUD model parameterized by $\mathbf{\Psi}$ yields mixtures of normal distributions associated with $\mathbf{u}_r$, and conditioned on the input channel state vector $\bar{\mathbf{h}}_r$,

$$p(\mathbf{u}_r|\bar{\mathbf{h}}_r;\mathbf{\Psi}) = \sum_{c=1}^{C_{\text{mix}}} \omega_{\mathbf{\Psi},c}\mathcal{N}\left(\mathbf{u}_r;\boldsymbol{\mu}_{\mathbf{\Psi},c}(\bar{\mathbf{h}}_r),\boldsymbol{\sigma}^2_{\mathbf{\Psi},c}(\bar{\mathbf{h}}_r)\right), \tag{3.2}$$

where C_{mix} is the total number of mixture components and, $\omega_{\mathbf{\Psi},c}$, $\boldsymbol{\mu}_{\mathbf{\Psi},c}$, and $\boldsymbol{\sigma}^2_{\mathbf{\Psi},c}$ being the mixture weight, means, and variances of the $c-$th Gaussian mixture, respectively. We treat $x-$ and $y-$coordinates of $\mathbf{u}_r$ as independent and restrict to a diagonal covariance matrix. Still, arbitrary distributions can be approximated by using the contribution from multiple mixtures [105, 106]. Then, we take a maximum likelihood perspective (MLE) and aim to learn a model that infers the parameters $\boldsymbol{\mu}_{\mathbf{\Psi},c} \in \mathbb{R}^2$ and $\boldsymbol{\sigma}^2_{\mathbf{\Psi},c} \in \mathbb{R}^2$ that maximize the likelihood of observing the desired location, $\mathbf{u}_r$. This is achieved by minimizing the negative log-likelihood (NLL) as

$$\mathcal{L}_{nll}(\mathbf{u}_r,\bar{\mathbf{h}}_r,\mathbf{\Psi}) := \sum_{r=1}^{R} -\log \sum_{c=1}^{C_{\text{mix}}} \omega_{\mathbf{\Psi},c}\mathcal{N}\left(\mathbf{u}_r;\boldsymbol{\mu}_{\mathbf{\Psi},c}(\bar{\mathbf{h}}_r),\boldsymbol{\sigma}^2_{\mathbf{\Psi},c}(\bar{\mathbf{h}}_r)\right). \tag{3.3}$$

In (3.3), parameters $\{\omega_{\mathbf{\Psi},c},\boldsymbol{\mu}_{\mathbf{\Psi},c},\boldsymbol{\sigma}^2_{\mathbf{\Psi},c}\}_{=1}^{C_{\text{mix}}}$ become the outputs of the network and depend on the input channel, $\bar{\mathbf{h}}_r$.

The parameters of GMM must satisfy certain constraints, which have to be incorporated accordingly in the DNN [105]. Therefore, the last layer of the network outputs the weights, means, and variances as follows. To satisfy $\sum_{c=1}^{C_{\text{mix}}} \omega_{\mathbf{\Psi},c} = 1$ and output the probability values corresponding to the weights of the mixture in the range of $0 \leq \omega_{\mathbf{\Psi},c} \leq 1$, the output for this part is modelled with *softmax* activation as

$$\omega_{\mathbf{\Psi},c} = \frac{\exp\left(o^{\omega}_{\mathbf{\Psi},c}\right)}{\sum_{c'=1}^{C_{\text{mix}}}\exp\left(o^{\omega}_{\mathbf{\Psi},c'}\right)}, \tag{3.4}$$

where $o^{\omega}_{\mathbf{\Psi},c}$ corresponds to the input of the activation function of the neuron in the output layer for this part. Likewise, a *softplus* activation function is adopted to

satisfy the variance constraint, i.e., $\boldsymbol{\sigma}^2_{\boldsymbol{\Psi},c} \geq 0$,

$$\boldsymbol{\sigma}^2_{\boldsymbol{\Psi},c} = \log\left(1 + \exp\left(\mathbf{o}^{\sigma^2}_{\boldsymbol{\Psi},c}\right)\right), \tag{3.5}$$

where $\mathbf{o}^{\sigma^2}_{\boldsymbol{\Psi},c}$ denote the inputs of the activation function of units for the part of variance. For the means, we simply model it using an identity function, i.e., $\boldsymbol{\mu}_{\boldsymbol{\Psi},c} = \boldsymbol{o}^{\mu}_{\boldsymbol{\Psi},c}$. Similarly, $\mathbf{o}^{\boldsymbol{\mu}}_{\boldsymbol{\Psi},c}$ are the inputs of the activation function for each neuron in the output layer for the means. Motivated from the models based on the mixture of experts (MoE), where the $c-$th model is considered an expert for certain input space [107], we choose the final estimate as the mean $\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\Psi}}$, and variance $\widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}}$, corresponding to the highest weight mixture, $\max_{c \in C_{\text{mix}}} \omega_{\boldsymbol{\Psi},c}$. Here, $\widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}}$ corresponds to data uncertainty, $\boldsymbol{\sigma}^2_{data} = \widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}}$.

3.2.2 Model Uncertainty

While modelling the parameters of a distribution function can capture the data uncertainty, this does not allow us to gauge model (epistemic) uncertainty, i.e., the uncertainty over the parameters $\boldsymbol{\Psi}$. In order to output the confidence of the model, we next discuss two different approaches.

Monte Carlo with dropout First, we consider a Bayesian perspective similar to [102–104] to propagate the model uncertainty to the output of the network by placing a distribution over the parameters of the network. In this case, the goal is to utilize the posterior distribution $p(\boldsymbol{\Psi}|\mathcal{D})$. We approximate the intractable distribution with Monte Carlo (MC) based methods [103, 108]. We know from [103] that applying dropout during the test time is equivalent to performing variational inference with a Bernoulli distribution. This approximation is given as

$$p(\boldsymbol{\Psi}|\mathcal{D}) \approx q(\boldsymbol{\Psi}; \Phi) = \text{Bern}(\boldsymbol{\Psi}; \Phi_{\text{drop}}), \tag{3.6}$$

where Φ_{drop} is the dropout rate on the network weights at each layer. Thus, we perform S_{mc} stochastic forward passes with dropout at test time on the same input. The mean, as well as the total variance, are evaluated as follows:

$$\widehat{\mathbf{u}}_r = \widehat{\boldsymbol{\mu}}^{(\text{MCD})}_r = \frac{1}{S_{\text{mc}}} \sum_{s=1}^{S_{\text{mc}}} \widetilde{\boldsymbol{\mu}}_{\boldsymbol{\Psi}^{(s)}}\left(\bar{\mathbf{h}}_r\right),$$

$$\widehat{\boldsymbol{\sigma}}^{2(\text{MCD})}_{r,total} = \underbrace{\frac{1}{S_{\text{mc}}} \sum_{s=1}^{S_{\text{mc}}} \widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}^{(s)}}\left(\bar{\mathbf{h}}_r\right)}_{\widehat{\boldsymbol{\sigma}}^2_{data}} + \underbrace{\frac{1}{S_{\text{mc}}} \sum_{s=1}^{S_{\text{mc}}} \left(\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\Psi}^{(s)}}\left(\bar{\mathbf{h}}_r\right) - \widehat{\boldsymbol{\mu}}^{(\text{MCD})}\right)^2}_{\widehat{\boldsymbol{\sigma}}^2_{model}}. \tag{3.7}$$

In (3.7), $\hat{\mathbf{u}}_r = \widehat{\boldsymbol{\mu}}_r^{(\mathrm{MCD})}$ refers to the mean location estimate with dropout and $\widehat{\boldsymbol{\sigma}}_{r,total}^{2(\mathrm{MCD})}$ refers to the associated total variance.

Deep ensemble For this MC-based method discussed above, the inference computation time scales linearly with the number of collected weights, S_{mc}. Therefore, we also evaluate another effective alternative to estimate the model uncertainty by sampling from an ensemble of S_{mc} different neural networks trained with S_{mc} randomly initialized sets of weights of the same network architecture. We refer to this as a LUD with a deep ensemble network (DEN). Similar to the dropout-based approach, we obtain the empirical mean $\widehat{\boldsymbol{\mu}}^{(\mathrm{DEN})}$ and total variance $\widehat{\boldsymbol{\sigma}}_{total}^{2(\mathrm{DEN})}$ of the distribution of location estimates. While for training, we require S_{mc} different independent trained models and sets of parameters to be stored, we only use a single forward pass during inference.

The variations of the Location and Uncertainty-Aware DNN (LUD) model we detailed in the last two sections allow us to address different types of uncertainties: they can specifically address either data-related uncertainty, model-related uncertainty, or simultaneously account for both. Finally, the two versions of acquiring model uncertainty allow us to trade between computational complexity and memory, too.

Model Architecture

The network architecture of LUD is depicted in Fig. 3.2. It uses $L = 4$ hidden layers from base DNN and ReLU activation functions for layers $l = \{1, 2, 3, 4\}$. We model the output layer as described in Section 3.2.1, which returns $\widehat{\boldsymbol{\mu}}_c^{(\cdot)}$, $\widehat{\boldsymbol{\sigma}}_c^{(\cdot)}$ and $\widehat{\omega}_c^{(\cdot)}$. The size of the output layer is $5C_{\mathrm{mix}} = 15$ units. For LUD with stochastic dropout approach (MCD), we place the dropout after $l = \{1, 2, 3\}$ of the network and search over a grid for $\Phi_{\mathrm{drop}} \in \{0.05, 0.1, 0.2\}$. In most instances, a dropout value of 0.1 yields good results for both location and uncertainty estimates, and is therefore chosen for the presented experiments in the next section. We train the model for 600 epochs with Adam solver [109], batch size of 512 at a fixed learning rate of 10^{-3}, and early stopping if validation loss is not reduced for 80 consecutive epochs. Weights are initialized from $\mathcal{N}(0, 10^{-2})$. For the LUD with an ensemble approach for model uncertainty (i.e., DEN), we train all individual networks for 300 epochs without dropout, thus faster converge time. However, to regularize the training process, in addition to early-stopping after 30 epochs, we clip the gradients at a value of 1.0. Other parameters are kept the same as for MCD.

3.2.3 Datasets and Quality of Estimates

In the following, before describing the simulation results, we first outline the datasets and metrics used for evaluating the quality of estimates.

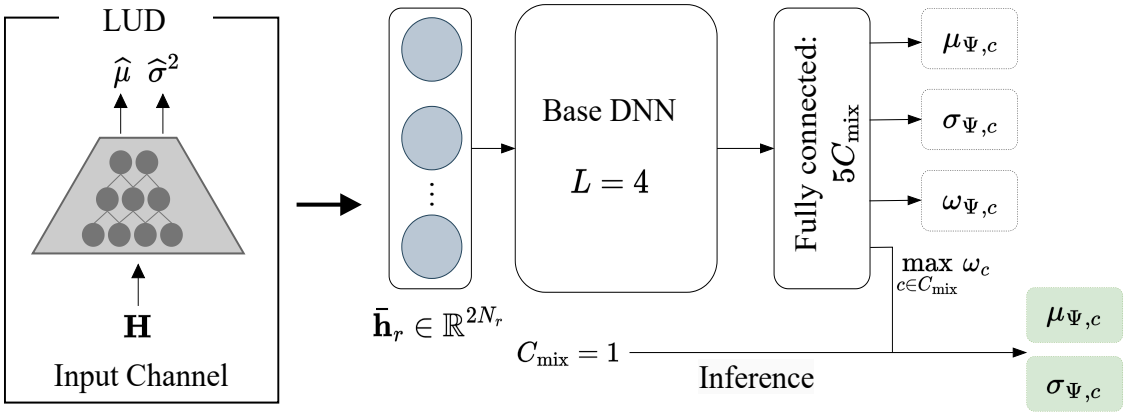


Figure 3.2: Model architecture overview of Location and Uncertainty-Aware DNN (LUD). During the training, the network learns a full parametric Gaussian mixture model (GMM) over locations. During the localization phase, the optimal location estimate is considered from the largest weight mixture with the associated variance.

Datasets

In this chapter, we rely on publicly available ray-tracing-based datasets introduced in [110] to evaluate the localization performance of LUD in a co-located massive MIMO setup. More specifically, we use CSI acquired from two distinct scenarios that depict an outdoor and an indoor environment:

1) *The outdoor scenario* of our interest is denoted as O1_3p5B in [110], and illustrated in Fig. 3.3. The scenario has users in LOS as well as NLOS. NLOS user locations are blocked by a metal screen which is placed in front of the BS. Two reflecting objects, represented in the figure by two other vehicles are also present.

2) *The indoor scenario*, denoted as I3_2p4 in reference [110], consists a mix of LOS and NLOS UEs too. An illustration of this scenario is provided in Fig. B.1 of Appendix B.

We consider $L_{\text{path}} = 5$ paths which may include both direct LOS and NLOS components, the latter comprising reflected paths, potentially involving multiple bounces from the buildings in the surrounding. At the BS, we consider a uniform planar array with $N_r = 16 \times 8$ antenna elements aligned along the $x-$ and $y-$axis. The region considered for the outdoors scenario (O1_3p5B) is R800−R1200. R800−R1200 is the default notation in [110] used to express the rows in a grid layout. Each row has $R' = 181$ user locations, and all users in this region are served by BS named BS−3 in the dataset. Table 3.1 summarizes the simulation parameters.

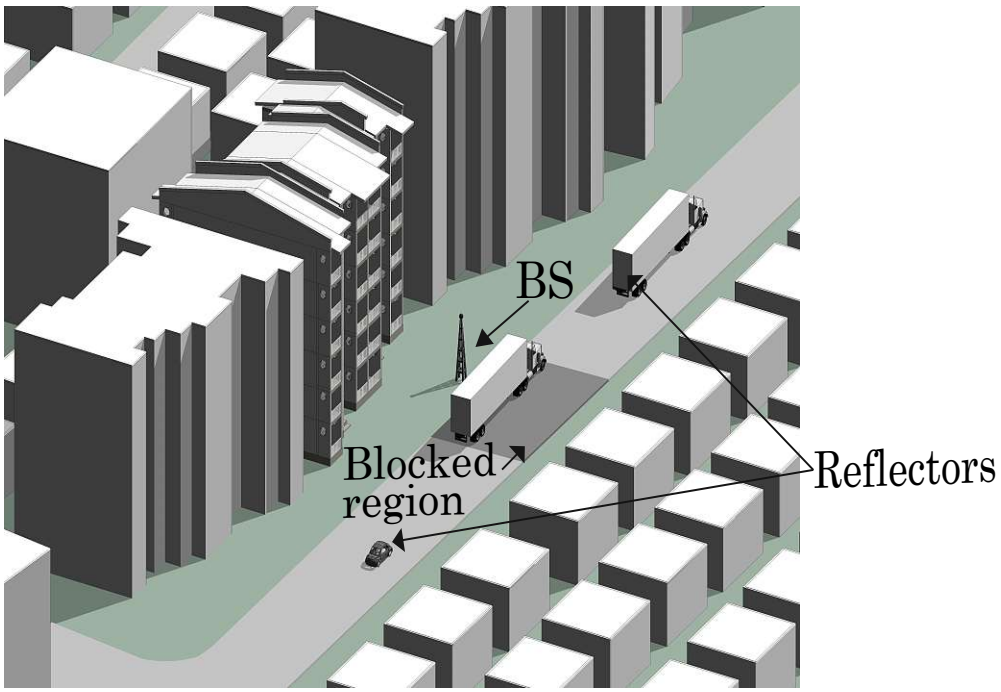


Figure 3.3: Illustration of an outdoor scenario considered for evaluation.

Quality of Estimates

The performance of the location estimation is quantified using the RMSE. Furthermore, to assess the quality of the acquired uncertainty estimates, we introduce a comparative analysis between these estimates and the ground-truth error. Specifically, we focus on ordering locations defined by the uncertainty estimates (referred to as 'confidence') and compare this with the ordering defined by actual location estimation errors (referred to as 'oracle'). The oracle-based ordering, denoted by $\text{RMSE}_{\text{orac}}$ represents the RMSE values from the ground-truth data. In contrast, the confidence-based ordering, denoted as $\text{RMSE}_{\text{conf}}$, is derived from the uncertainty estimates provided by the model. Intuitively, if we remove locations with higher uncertainty (as indicated by the model), we expect the RMSE of the remaining locations to be lower, indicating more accurate estimations. Therefore, we evaluate their difference, i.e., the error between the ordering of locations defined by RMSE (*oracle*) and the ordering defined by the uncertainty estimates (*confidence*),

$$\alpha_{R_i} = \text{RMSE}_{\text{orac}}(b_{R_i}) - \text{RMSE}_{\text{conf}}(b_{R_i}), \tag{3.8}$$

where b_{R_i}, in this case, represents the fraction of removed locations based on their ranked uncertainty. A smaller value of α_{R_i} suggests a higher *alignment* between the model's uncertainty estimates and the actual errors.

Table 3.1: Parameters for the investigated scenarios.

	Outdoor scenario (O3_2p5)	Indoor scenario (I3_2p4)
Frequency, f_c	3.5 GHz	2.4 GHz
Bandwidth, B_{sys}	20 MHz	20 MHz
BS Number	BS−3	BS−2
Number of paths, L_{path}	5	5
Subcarriers, N_c	1024	1024
User locations	R800−R1200	R1−R1159

Furthermore, to compare the effectiveness of two variations of the LUD method and the influence of the MC samples, we evaluate the area under the confidence-oracle curve (AUCO) metric. This metric measures the performance across multiple thresholds of removed locations and provides an aggregate score. The smaller AUCO value, the better-acquired uncertainty explains the variations in locations with respect to RMSE.

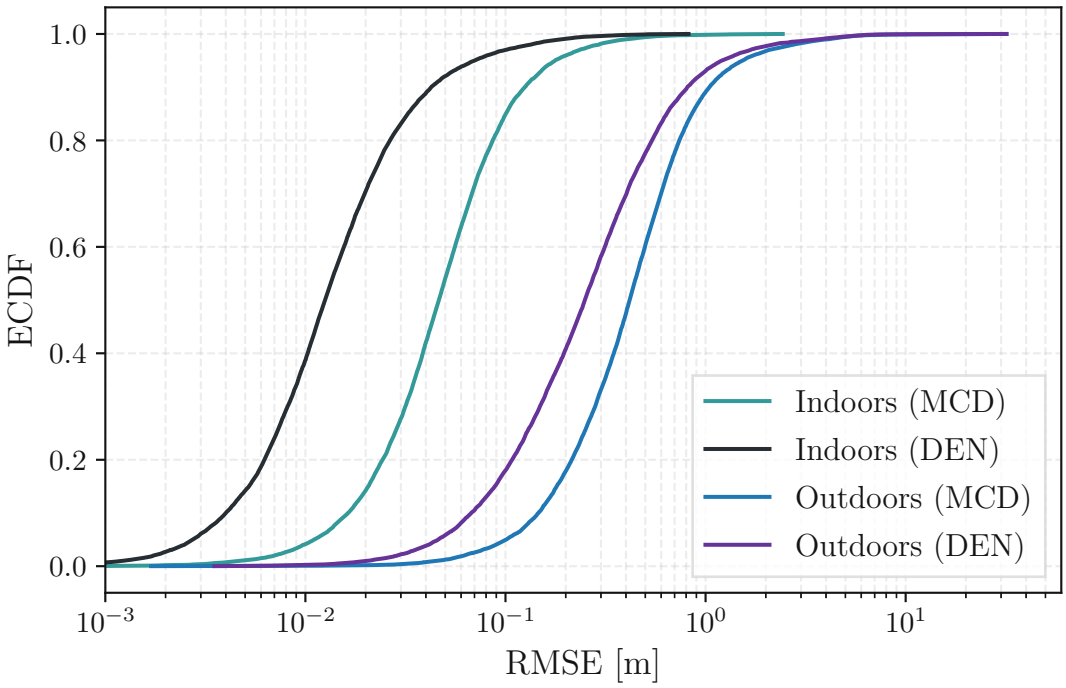
Localization Accuracy

Localization accuracy in terms of RMSE for the two approaches and all reference scenarios is depicted in Fig. 3.4. Both methods can achieve better than 1.5m accuracy at 90-th percentile in the outdoor scenario, and less than 20cm indoors. We observe in Fig. 3.4b), 3.4c) that accuracy improves with S_{mc}. Averaging over S_{mc} different weight configurations has a pronounced positive impact on the overall RMSE. For the sake of comparison, we also provide results for $S_{\text{mc}} = 1$. This is equivalent to only estimating $\boldsymbol{\sigma}^2_{data}$ with the base DNN. DEN variation of LUD outperforms the MCD one.

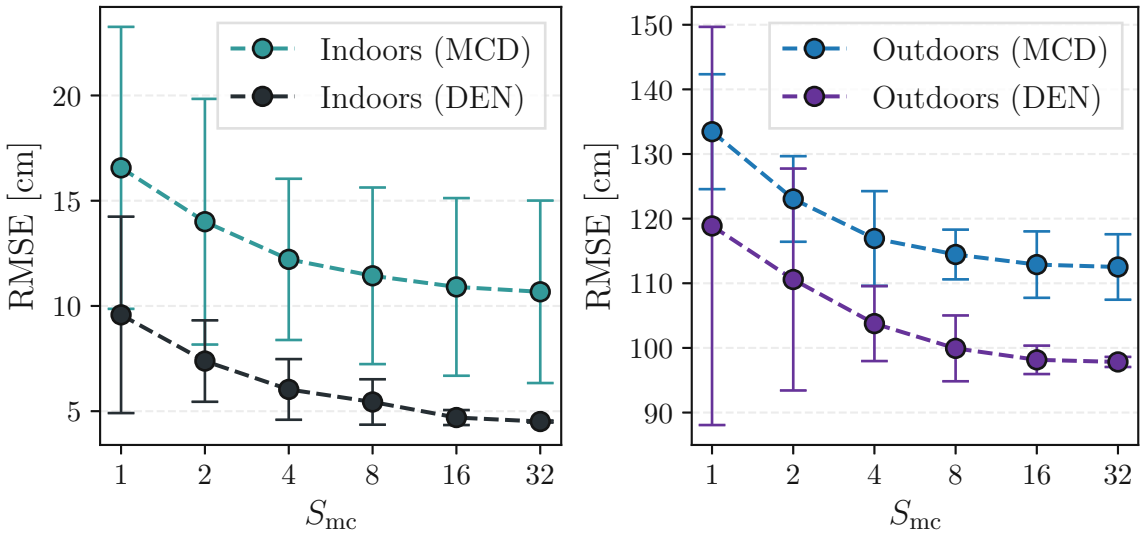
Uncertainty Accuracy

We quantify the acquired uncertainty in terms of AUCO as described in Section 3.2.3. If the estimated uncertainty well represents the variance, removing locations with the highest uncertainty in the test dataset should lead to a lower RMSE too. Thus, the confidence curve should be able to capture the variations in the RMSE. The oracle curve can also be interpreted as the lower bound, or ground-truth on uncertainty values that can be acquired by the model in a respective dataset. Ideally, both curves would match.

In Fig. 3.5a we plot normalized $\text{RMSE}_{\text{orac}}$ and $\text{RMSE}_{\text{conf}}$. In Fig. 3.5b we plot the

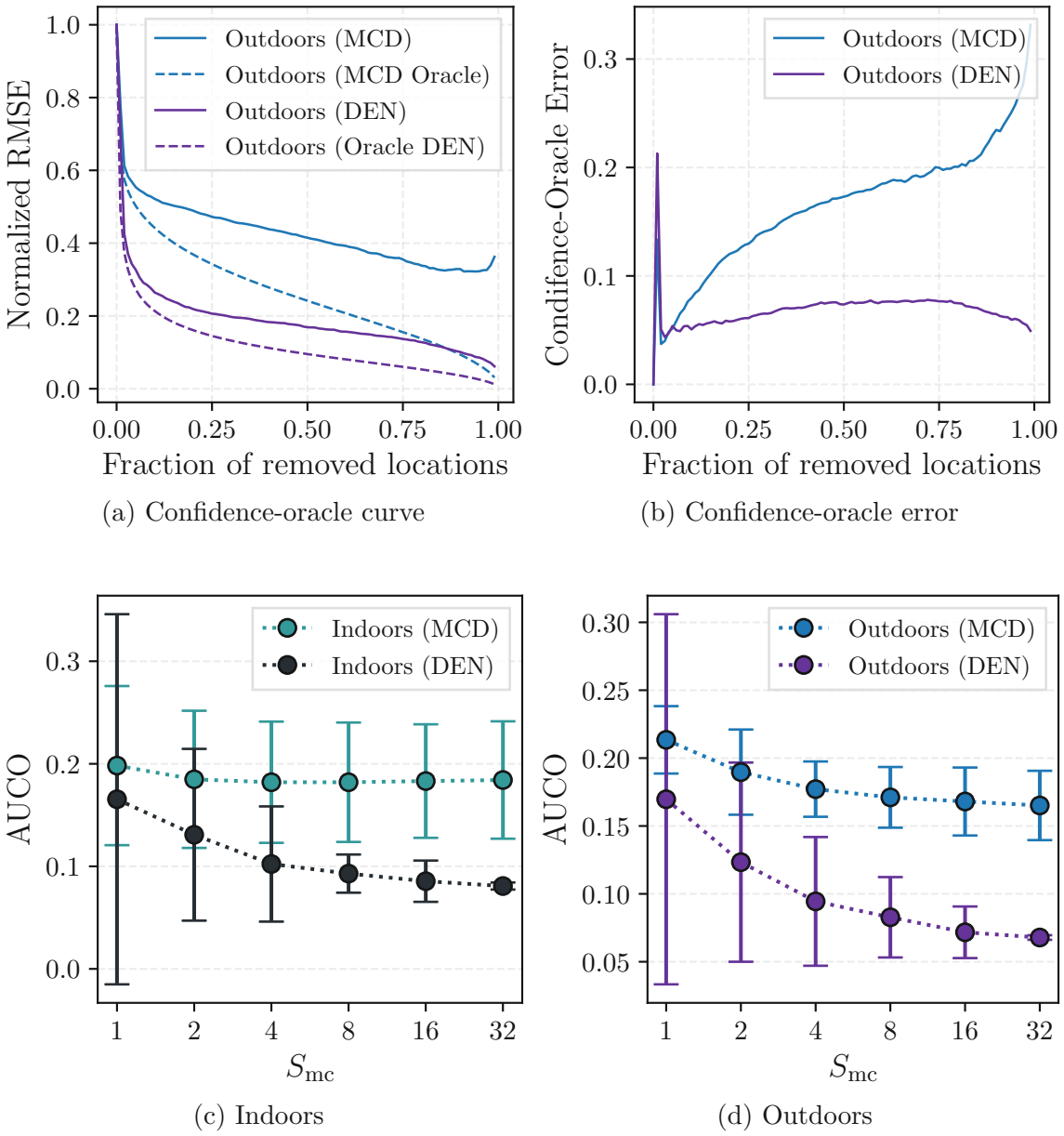


(a) Indoors vs Outdoors for LUD with DEN and MCD.



(b) Indoors (c) Outdoors

Figure 3.4: Localization error for a) indoors and outdoors when $S_{\text{mc}} = 32$. Accuracy improves for $S_{\text{mc}} > 1$ for both indoors c) and outdoors d) scenarios.



(a) Confidence-oracle curve

(b) Confidence-oracle error

(c) Indoors

(d) Outdoors

Figure 3.5: Evaluation of variance based ordering for uncertainty estimation. Example of confidence and oracle curves a) and error between the confidence and oracle curve b) for $S_{\text{mc}} = 32$. LUD with DEN can better explain the variations for both large and small RMSE values.

confidence-oracle error, i.e., the difference between $\text{RMSE}_{\text{orac}}$ and $\text{RMSE}_{\text{conf}}$ as in (3.8). In both figures, we illustrate that a deep ensemble can better capture the variations in the RMSE. The evaluation in terms of AUCO is depicted in Fig. 3.5c and Fig. 3.5d for indoor and outdoor scenarios, respectively. We can easily notice

that the ensemble performs better than stochastic dropout approach, and the gap becomes more evident as $S_{\text{mc}} > 4$.

The observed inferior performance of the stochastic dropout approach, in contrast to the ensemble approach, can be attributed to its sensitivity to the dropout rate. This hyperparameter is a critical aspect of the model's performance, yet the optimal selection of the same is challenging. Finally, in Fig. 3.5a, we can observe that by removing 20% of locations with the highest error, the overall accuracy improves by 80% for an ensemble-based approach in this outdoor scenario.

Impact of Data and Model Uncertainties

In Fig. 3.6, we provide qualitative results for uncertainty estimation for the two cases: NLOS and out-of-set region. We plot normalized values for the RMSE as well as for the total uncertainty estimates as evaluated in (3.7). Additionally, in this figure, we separately plot the data uncertainty and model uncertainty, denoted by $\widehat{\boldsymbol{\sigma}}^2_{data}$ and $\widehat{\boldsymbol{\sigma}}^2_{model}$ in (3.7).

Specifically, we consider the outdoor scenario, where locations for users behind the blockage have the highest RMSE; consequently, LUD should output high uncertainty too. Moreover, our goal is also to understand if model uncertainty can improve our awareness about out-of-set regions. Therefore, we remove all training samples from a region marked in a blue rectangle box as the out-of-set region in Fig. 3.6. Likewise, we expect the LUD to generate high uncertainty score for users in the out-of-set region during the testing phase.

Finally, we can observe that the data uncertainty is sufficient to capture the error when users are in deep NLOS, i.e., just behind the blockage. However, out-of-set cases are more challenging and acquiring model uncertainty enhances the overall dependability.

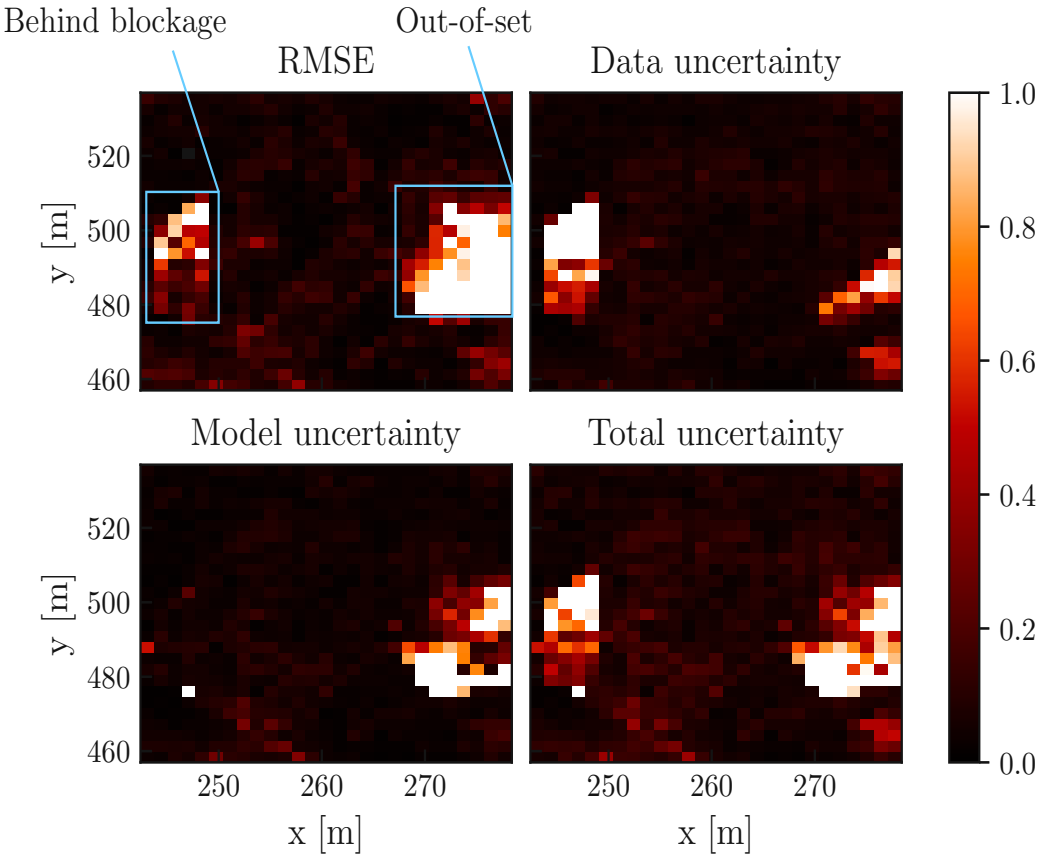


Figure 3.6: The impact of data and model uncertainty estimation in NLOS and out-of-set case. Lighter-colored regions indicate higher error and uncertainty. The figure shows the case of LUD with ensemble approach.

3.3 DAS and RRH Subset Selection

In the first part of this chapter, we focused on evaluating the localization accuracy in a co-located massive MIMO BS. However, we can apply the same approach to locate the user from CSI obtained in a DAS. Due to DAS achieving greater spatial diversity, they can provide substantial advantages for wireless positioning systems. Hence, in this part, we investigate LUD in a DAS and aim to understand the impact of NLOS strength. Regardless, a primary limitation of DAS is the necessity for costly infrastructure to establish connections between RRHs and CU. Moreover, acquiring signal information from a dense number of RRHs may increase the fronthaul overhead and computational burden at the CU. Hence, a selection strategy that leverages CSI solely from the "best" K_{rsd} RRHs out of the total M RRHs is desirable, with a minimal impact on localization performance. Therefore, after analyzing the localization performance in a DAS, we investigate whether a learned subset of RRHs can maintain good localization accuracy. Accordingly, our focus will shift towards extending the LUD to a learning-based approach capable of RRH subset selection, combined with user location and uncertainty estimation.

To assess the localization accuracy, we utilized a channel model based on ray-tracing in the initial portion of this chapter. In contrast, in the remaining part, we model the signal information within a DAS slightly differently. Since we aim to consistently evaluate its performance against a co-located antenna setup and gain better control in understanding the impact of NLOS signal strength, we utilize the channel model proposed in [111]. The subsequent section will outline the signal information used for the remaining investigation.

3.3.1 Signal Information in DAS

We consider a distributed antenna system as shown in Fig. 3.7. We assume that all RRHs are connected via high-speed fronthaul links to the CU, i.e., the delay between the RRHs and the CU is negligible. Initially, each RRH obtains signal information from R individual locations during the training phase and forwards it to the CU. The role of the CU is to process the per-user channel estimates and learn a subset of K_{rsd} RRHs. Then, throughout the operations phase, the role of the CU becomes to perform positioning inference of the unknown user transmitter with the channel information gathered from the subset of RRHs, which are learned during the preceding phase. Instead of reusing the same geometric channel model and the signal information detailed in Chapter 2, here we consider that the channel between the $r-$th user location and the $m-$th RRH is [111]

$$h_{r,m} = \alpha_{r,m} + \sum_{g=1}^{G} \beta_{r,g}\alpha_{g,m} \quad , \tag{3.9}$$

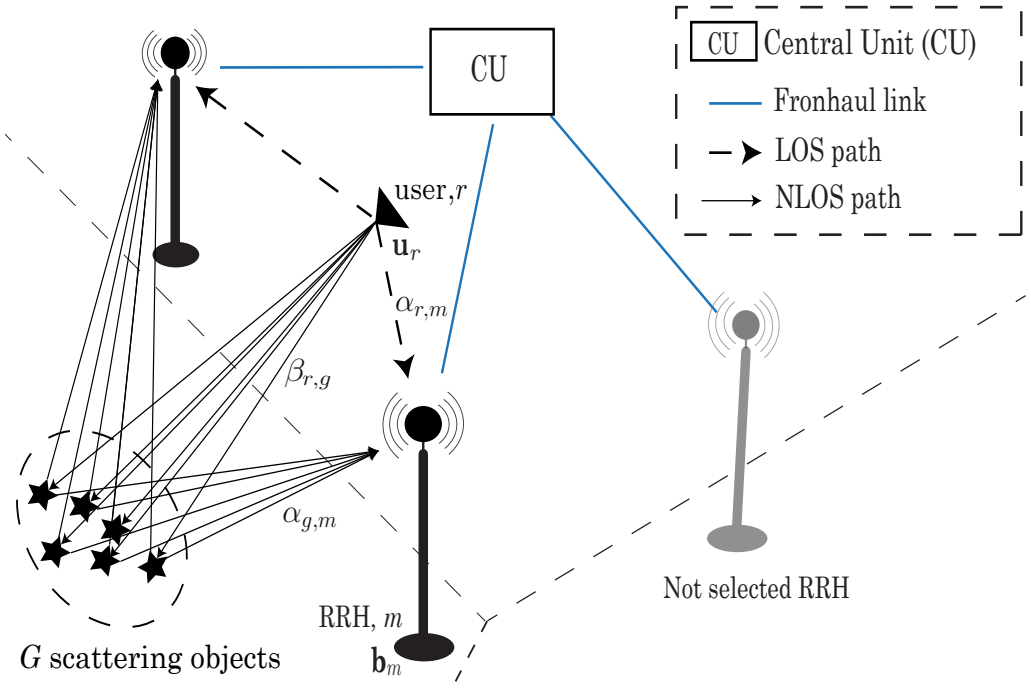


Figure 3.7: Illustration of the system model considered for RRH selection and user location estimation in a distributed antenna system (DAS).

where $\alpha_{r,m}$ is the direct line-of-sight (LOS) path, i.e., it describes the propagation without involving any scattering. The sum term describes the signal over G other paths that include a scattering event due to the scattering objects placed in the geographical ROI. As in [111], path coefficients $\alpha_{i,j}$ and $\beta_{i,j}$ are obtained as:

$$\begin{aligned} \alpha_{i,j} &= \frac{\lambda}{4\pi\|\mathbf{x}_i-\mathbf{x}_j\|} e^{i\frac{2\pi}{\lambda}\|\mathbf{x}_i-\mathbf{x}_j\|}, \\ \beta_{i,j} &= \frac{\delta_j}{\sqrt{4\pi}\|\mathbf{x}_i-\mathbf{x}_j\|} e^{i\frac{2\pi}{\lambda}\|\mathbf{x}_i-\mathbf{x}_j\|}. \end{aligned} \tag{3.10}$$

The coefficients in (3.10) describe the wave propagation characteristics between any two elements i and j at their respective positions $\mathbf{x}_i$ and $\mathbf{x}_j$ in the ROI. The scattering event δ_j captures the electromagnetic properties of the scattering objects, and λ denotes the carrier wavelength. We assume that scattering objects impose a random phase-shift, φ_j, of the signal and have an amplitude gain of $\gamma_j = |\delta_j|$. The amplitude gain is equal to the square root of the bi-static radar cross section (RCS) of the scattering object. As shown in [111], it can be related to the Rician K-factor of the effective multipath channel and, therefore, determines the relative strength of LOS and NLOS components. Consequently, we are able to model a more realistic propagation environment that distinguishes from the extreme LOS and non-line-of-sight (NLOS) cases in order to better understand the performance limits

of the methods in terms of localization error. The resulting channel model, after substituting the scattering coefficients (3.10) and assuming $\gamma_j = \gamma, \forall j$, is given as

$$h_{r,m} = \frac{\lambda}{4\pi \left\|\mathbf{u}_r - \mathbf{b}_m\right\|} e^{i\frac{2\pi}{\lambda}\left\|\mathbf{u}_r - \mathbf{b}_m\right\|} + \frac{\lambda\gamma}{(4\pi)^{\frac{3}{2}}} \sum_{g=1}^{G} u\left(\mathbf{p}_g\right). \tag{3.11}$$

The individual sum terms are further expressed as

$$u\left(\mathbf{p}_g\right) = \left(\left\|\mathbf{u}_r - \mathbf{p}_g\right\| \left\|\mathbf{p}_g - \mathbf{b}_m\right\|\right)^{-1} \times e^{i\left(\varphi_k + \frac{2\pi}{\lambda}\left(\left\|\mathbf{u}_r - \mathbf{p}_g\right\| + \left\|\mathbf{p}_g - \mathbf{b}_m\right\|\right)\right)}. \tag{3.12}$$

We maintain a minimum distance between any transmit antenna location and RRH, $\left\|\mathbf{u}_r - \mathbf{b}_m\right\| > 0$, as well as between any scattering object and RRH, $\left\|\mathbf{b}_m - \mathbf{p}_g\right\| > 0$. Finally, for each user r, the CU forms a $M \times 1$ channel vector $\mathbf{h}_r$, i.e.,

$$\mathbf{h}_r = \left[h_{r,1}, h_{r,2}, \ldots, h_{r,M}\right]^T \in \mathbb{C}^{M\times 1}. \tag{3.13}$$

3.3.2 Localization Accuracy in DAS

First, we investigate the localization accuracy of LUD in a DAS without any selection strategy. We setup a scenario where we consider $M \in \{16, 36, 64\}$ RRHs distributed in a grid configuration within the ROI of 100m $\times$ 100m range. An example of the simulation scenario with $M = 64$ is depicted in Fig. 3.8. We place $G = 100$ scattering objects normally distributed in clusters $i \in \{1, 2, 3\}$ with $\mathcal{N}(\boldsymbol{\mu}_g^{(i)}, \boldsymbol{\sigma}_g^{(i)})$, where $\boldsymbol{\mu}_g^{(1)} = [0, -60]$, $\boldsymbol{\mu}_g^{(2)} = [60, 0]$, $\boldsymbol{\mu}_g^{(3)} = [0, 60]$, $\boldsymbol{\sigma}_g^{(1)} = [100, 1]$, $\boldsymbol{\sigma}_g^{(2)} = [1, 100]$, and $\boldsymbol{\sigma}_g^{(3)} = [100, 1]$. The scenario chosen for our numerical experiments represents a public space, surrounded on three sides by buildings. The scattering coefficient is set as $\gamma = \{1.2, 3.0\}$ to provide insights on the impact of NLOS components of the multipath channel. We uniformly sample from $R = 49\,000$ locations in the ROI and we set the minimum distance $\left\|\mathbf{u}_r - \mathbf{b}_m\right\| > 0.5$m.

Impact of NLOS

We look into the impact of the number of RRHs as the NLOS components become stronger. This is shown in Fig. 3.9. We can observe that increasing γ, i.e., the value of scattering coefficients, the performance degrades in terms of the localization error for approximately 50% at 90−th percentile in case of $M = 16$. Naturally, increasing the number of RRHs improves the overall performance but can also compensate for the impact of the dominant NLOS components, where the gap of the error at 90−th percentile is approximately 15% for $M = 64$.

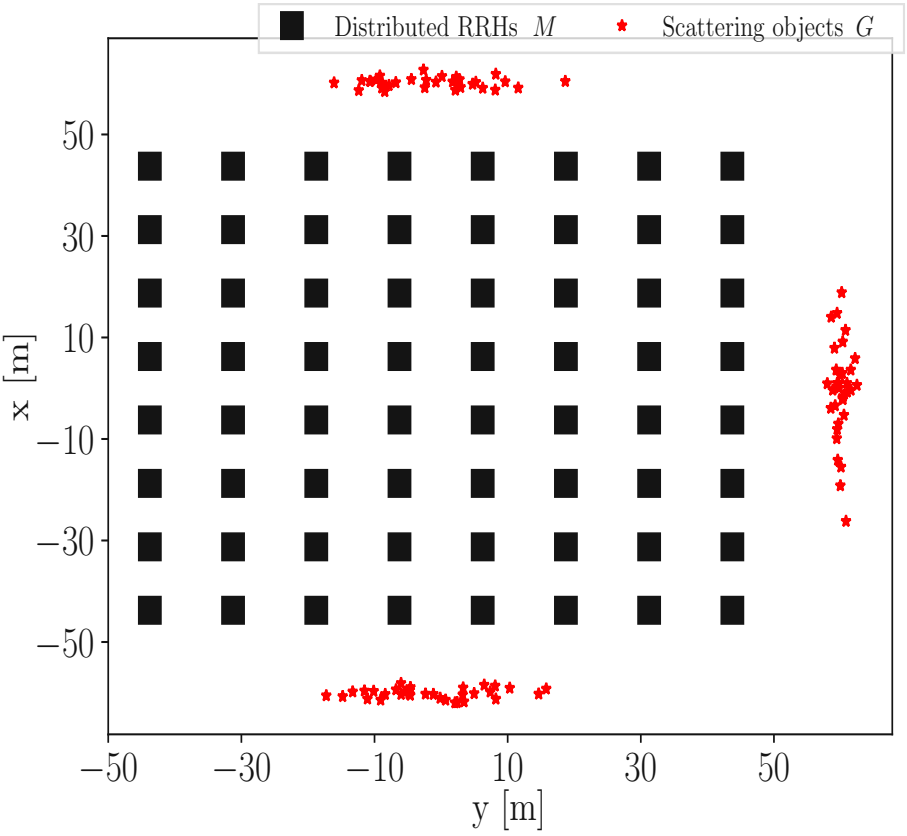


Figure 3.8: Simulation setup with M spatially distributed RRHs.

DAS versus Co-located Setup

Next, we compare the localization accuracy of a DAS with centralized massive MIMO, i.e., $M = 64$ co-located antennas. In the case of co-located setup, we consider that antennas are aligned in a circular array with a diameter of 1.5m. As in the first part of this chapter, the channel vector considered for comparison is $\widehat{\mathbf{h}}_r \in \mathbb{R}^{2N_r}$. In Fig. 3.10, we can observe that DAS improves the performance by more than 30% at 90−th percentile. Further, increasing the number of antenna elements does not lower the error for the centralized case.

3.3.3 Remote Radio Head Selection

In this section we aim to reduce the computational complexity at the CU by determining which RRHs should be selected while providing accurate and reliable localization performance. The selection process is fixed and performed once per ROI. The selected RRHs are used for all the test users over a period of time until the next

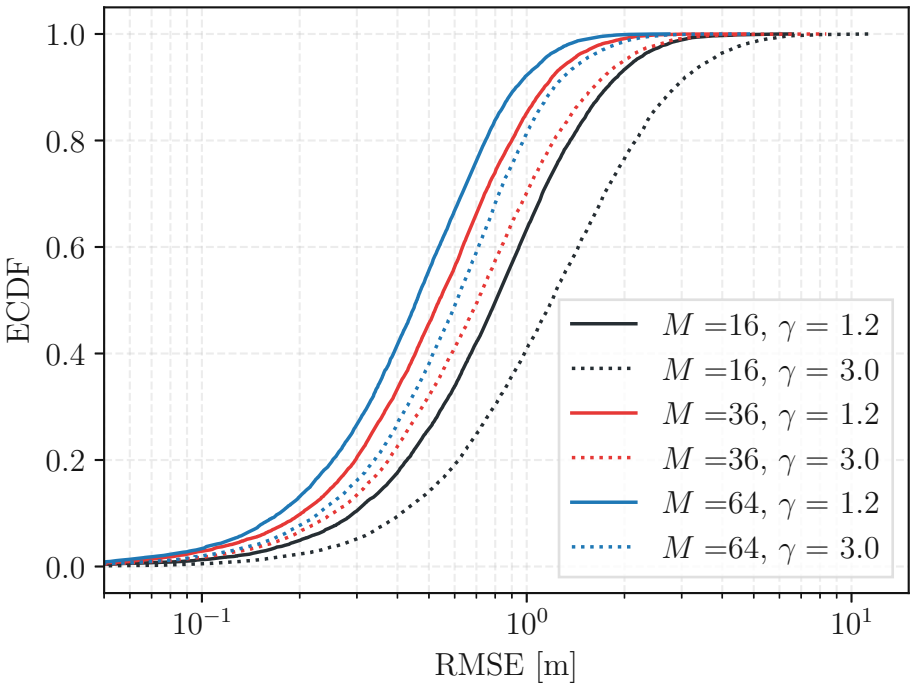


Figure 3.9: Localization accuracy in DAS for different values of M and $\gamma = \{1.2, 3.0\}$.

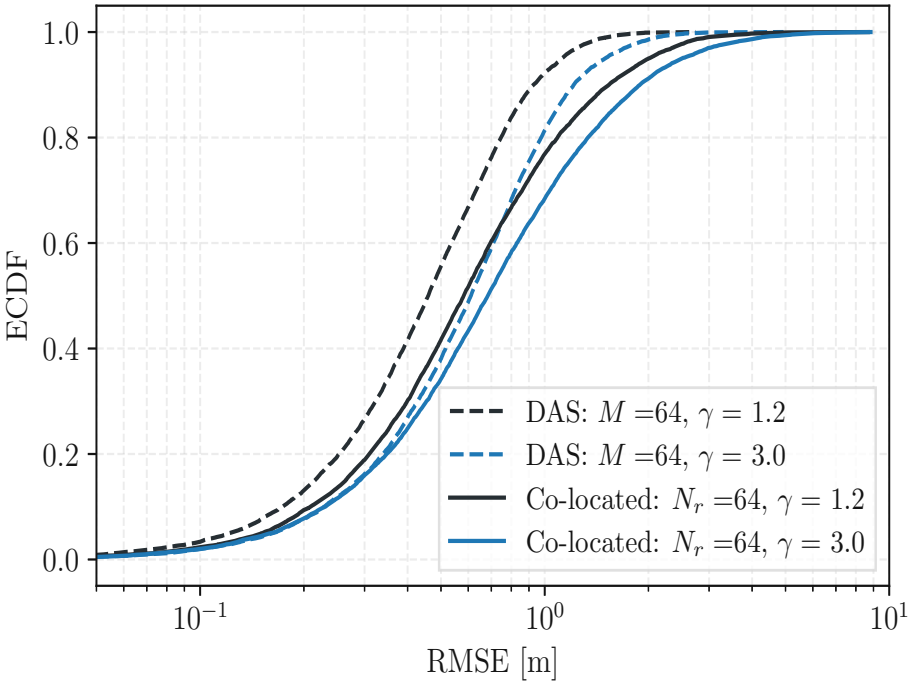


Figure 3.10: ECDF - DAS vs co-located massive MIMO.

scheduled selection process occurs. We approach this problem in two steps. First, we describe the developed end-to-end and DNN-based RRH selection method, which we will refer to as RRH subset selection DNN (RSD). Then, using the subset of RRHs, we elaborate on how to utilize LUD to estimate the final position of the transmitter and the uncertainty associated with it. The RRH selection and localization process is shown in Fig. 3.11. The RSD relies on channel strength information as an input, $\widetilde{\mathbf{h}}_r = |\mathbf{h}_r|$. Similar to the first part of the chapter, the dataset is scaled by dividing the inputs with the maximum absolute value in it, $\Delta = \max(\{|h_{r,1}|, \ldots, |h_{r,M}|\}_{r=1}^{R})$.

Specifically, our goal is to learn a subset $\mathcal{K}_{\text{rsd}} \subseteq \mathcal{M}$ of spatially distributed RRH indices of $|\mathcal{K}_{\text{rsd}}| = K_{\text{rsd}}$, as well as a channel to location mapping function $f_{\boldsymbol{\Theta}}^{(\text{RSD})} : \mathbb{R}^{K_{\text{rsd}}} \mapsto \mathbb{R}^2$, where $\boldsymbol{\Theta}$ denotes the parameters (weights) of RSD. Thus, we seek to minimize the localization error between the estimated $\widehat{\mathbf{u}}_r = f_{\boldsymbol{\Theta}}^{(\text{RSD})}(\widehat{\mathbf{h}}_r)$ and the given location information $\mathbf{u}_r$. Formally, our goal becomes to learn the optimal $\mathcal{K}_{\text{rsd}}^{\star}$ and $\boldsymbol{\Theta}^{\star}$ such that

$$\begin{aligned} \{\mathcal{K}_{\text{rsd}}^{\star}, \boldsymbol{\Theta}^{\star}\} &= \underset{\mathcal{K}_{\text{rsd}}, \boldsymbol{\Theta}}{\arg\min}\, \mathcal{L}_{\text{RSD}}(\mathbf{u}_r, \widehat{\mathbf{h}}_r, \boldsymbol{\Theta}) \\ &= \underset{\mathcal{K}_{\text{rsd}}, \boldsymbol{\Theta}}{\arg\min}\, \mathbb{E}\left[\left\| f_{\boldsymbol{\Theta}}^{(\text{RSD})}(\widehat{\mathbf{h}}_r) - \mathbf{u}_r \right\|^2\right]. \end{aligned} \tag{3.14}$$

In (3.14), $\widehat{\mathbf{h}}_r$ contains CSI from K_{rsd} selected RRHs. It can be written as

$$\widehat{\mathbf{h}}_r = \mathbf{A}\widetilde{\mathbf{h}}_r \,, \tag{3.15}$$

where $\mathbf{A} \in \{0,1\}^{K_{\text{rsd}} \times M}$ represents a binary selection matrix with k selection-rows $\mathbf{a}_k$, i.e., $a_{k,m} = 1$ if $k \in \mathcal{K}_{\text{rsd}}$.

3.3.4 RSD - RRH Sampling and Selection

The RRH selection in (3.14), is a combinatorial optimization problem over the discrete set $\mathcal{K}_{\text{rsd}} \subseteq \mathcal{M}$ of RRHs. This implies that the number of RRH subsets grows exponentially with M and K_{rsd}. In addition, implementation difficulties arise due to the inability to backpropagate over the discrete variables. To circumvent these difficulties, we rely on a *reparameterization* approach [112]: instead of directly optimizing over $\mathbf{a}_k$, we relax it by a learnable $K_{\text{rsd}}-$dimensional vector $\boldsymbol{\phi}_k \in \mathbb{R}_{+}^{M}$. We do so by introducing an RRH selection layer, trained together with the rest of the network, that utilizes *concrete* variables [113, 114], based on a Gumbel distribution [115]. In this case, the RRH selection layer controls the intensity of relaxation of the elements in the binary matrix $\mathbf{A}$, using the layer temperature $\tau \in (0, \infty)$.

Formally, we begin by defining z_k to be a categorical variable over the domain $\{1, 2, \ldots, M\}$, which can be represented by a vector of class probabilities $\boldsymbol{\pi}_k \in \mathbb{R}^M$,

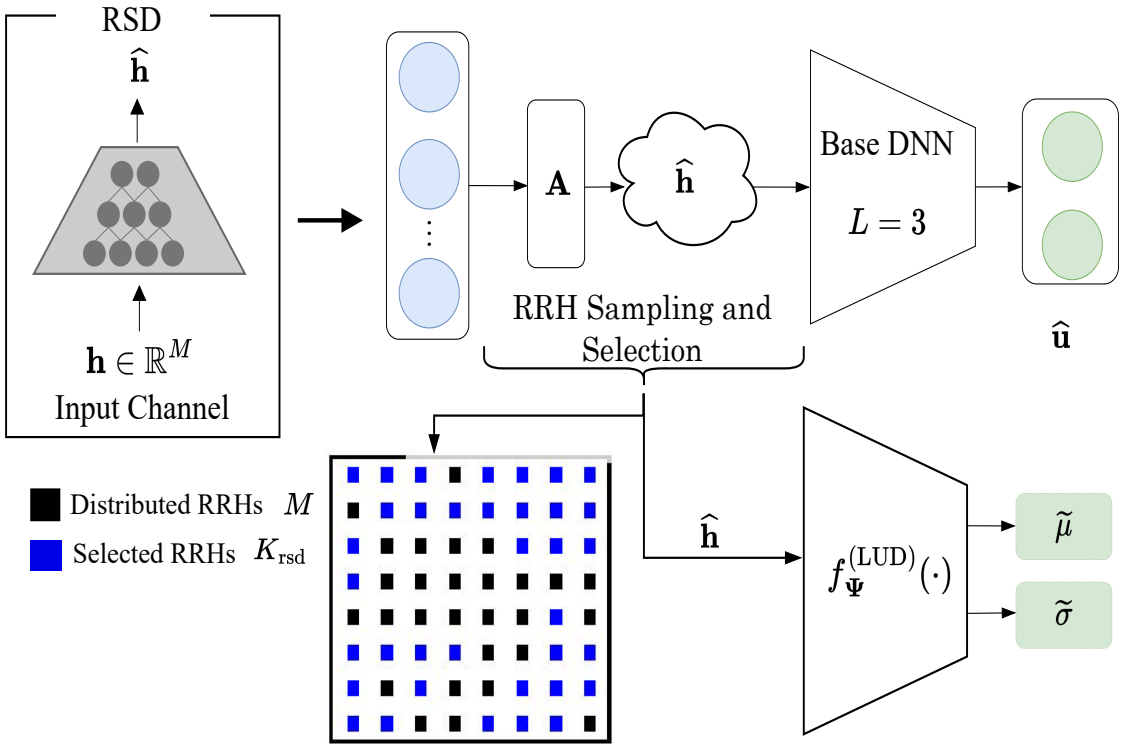


Figure 3.11: RRH subset selection DNN (RSD) with a sampling and selection layer and localization with uncertainty estimation DNN (LUD).

with the role of selecting $m-$th RRH in $k-$th row, i.e.,

$$z_k \sim \texttt{Cat}(\boldsymbol{\pi}_k, M), \tag{3.16}$$

where the class probabilities are related to vector $\boldsymbol{\phi}_k$ as

$$\pi_{k,m} = \frac{\exp \phi_{k,m}}{\sum_{i=1}^{M} \exp \phi_{k,i}}. \tag{3.17}$$

Using the Gumbel-Max trick [115, 116], we can sample from (3.16) by independently perturbing $\phi_{k,m}$ with i.i.d. Gumbel noise, $g_{k,m} = \texttt{Gumbel}(0, 1)$. Specifically, we draw M samples from a uniform distribution and form $\mathbf{q}_k \sim \mathcal{U}(0, 1)$. Then, we apply the `Gumbel` inverse CDF, $F_G^-(\mathbf{q}_k)$. In more detail,

$$\boldsymbol{g}_k = F_G^-(\mathbf{q}_k) = -\log(-\log(\mathbf{q}_k)) \tag{3.18}$$

and

$$\tilde{z}_k = \underset{m\in\{1,2,\ldots,M\}}{\arg\max} \{\phi_{k,m} + g_{k,m}\}. \tag{3.19}$$

The arg max of the respective perturbed probabilities returns a sample from the original categorical distribution, $\tilde{z}_k \overset{\mathrm{d}}{=} z_k$. Finally, during the forward pass, we can convert $\tilde{z}_k$ into a binary vector $\mathbf{a}_k$, such that $\|\mathbf{a}_k\| = 1$ and we write

$$\begin{aligned} \mathbf{a}_k &= \texttt{one\_hot}_M\{\tilde{z}_k\} \\ &= \texttt{one\_hot}_M\{\arg\max_m\{\phi_{k,m} + g_{k,m}\}\}. \end{aligned} \tag{3.20}$$

However, in order to allow gradient backpropagation during the training phase, we need to relax the non-differentiable $\arg\max\{\cdot\}$ operation. To do so, we rely on Gumbel-Softmax [114] and approximate the $\arg\max\{\cdot\}$ in (3.19) with a continuous soft operation, $\text{soft}\max_\tau\{\cdot\}$. Then, $\mathbf{a}_k$ is replaced by

$$\mathbf{a}_k = \lim_{\tau\to 0} \frac{\exp\left\{\tau^{-1}\left(\boldsymbol{\phi}_k + \boldsymbol{g}_k\right)\right\}}{\sum_{m=1}^{M}\exp\left\{\tau^{-1}\left(\phi_{k,m} + g_{k,m}\right)\right\}}. \tag{3.21}$$

As $\tau \to 0$, (3.21) smoothly approaches the arg max, and each of the neurons in the RRH sampling and selection layer outputs one of the selected RRHs. During the training over a total number of T_{epoch} epochs, the value of τ decreases exponentially, and for the $t-$th epoch, it is given by

$$\tau_{t_{\text{epoch}}} = \tau_0\left(\tau_{\text{e}}/\tau_0\right)^{t_{\text{epoch}}/T_{\text{epoch}}}, \tag{3.22}$$

where τ_0 and τ_{e} (and $\tau_{\text{e}} < \tau_0$) are the user-defined start and end temperature values, respectively. This enables the end-to-end network to initially explore various combinations of RRHs, gradually transitioning from a strategy of linearly combining the outputs of these RRHs to a strategy focused on selecting a particular RRH as training progresses. The iterative nature of the training process is designed to continue until a point of convergence is reached, where each unit in the selection layer consistently selects a single, and unique RRH, $k \in \mathcal{K}_{\text{rsd}}$.

Operation Phase

During the test time, i.e., the RRH selection process and localization, we drop the stochasticity of the network and replace the continuous $\text{soft}\max_\tau(\cdot)$ by a discrete $\arg\max(\cdot)$. The training as well operation phase of the RSD $\forall k \in \mathcal{K}_{\text{rsd}}$ are further summarized in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Training RSD

Given: A training dataset $\mathcal{D} = \{\mathbf{h}_r, \mathbf{u}_r\}_{r=1}^{R}$, number of RRHs to select K_{rsd}, epochs T_{epoch}, learning rate α, final temperature τ_{e}, start temperature τ_0

Build RSD, $f_{\boldsymbol{\Theta}}^{(\text{RSD})}(\cdot)$, and initialize $\boldsymbol{\Theta}$.

for $k \in \{1, 2, \ldots, K_{\text{rsd}}\}$ **do**
  Initialize a $M-$ dimensional vector $\boldsymbol{\phi}_m \in \mathbb{R}_+^M$.
end for
for $t_{\text{epoch}} \in \{1, 2, \ldots, T_{\text{epoch}}\}$ **do**
  Compute τ according to (3.22)
  for $k \in \{1, 2, \ldots, K_{\text{rsd}}\}$ **do**
    Compute $\mathbf{a}_k$ using (3.20)
    Set $\widehat{h}_{r,k} = \mathbf{a}_k \widetilde{\mathbf{h}}_r$
  end for
  Compute the gradients of $\mathcal{L}_{\text{RSD}}$ w.r.t. $\boldsymbol{\Theta}$ and each $\boldsymbol{\phi}_k$.
  Update the parameters $\boldsymbol{\Theta} \leftarrow \boldsymbol{\Theta} - \alpha \nabla_{\boldsymbol{\Theta}} \mathcal{L}_{\text{RSD}}$ and $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \alpha \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\text{RSD}}$
end for

Return: Trained $f_{\boldsymbol{\Theta}}^{(\text{RSD})}(\cdot)$ and learned parameters $\boldsymbol{\Theta}$ and $\boldsymbol{\phi}$.

Algorithm 2 Using RSD for RRH Selection

Given: A test channel $\widetilde{\mathbf{h}}_r$ and trained parameters $\boldsymbol{\phi}$

for $k \in 1, 2, \ldots, K_{\text{rsd}}$ **do**
  Compute $\tilde{z}_k = \arg\max_m \phi_{k,m}$
  Set $\widehat{h}_{r,k} = \widetilde{h}_{r,\tilde{z}_k}$
end for

Return: $\widehat{\mathbf{h}}_r$.

RSD Architecture Details

Similar to LUD, RSD employs ReLU for the intermediate non-linear operations. RSD consists of three intermediate layers, each containing 350 units. It only requires the channel gain as input, resulting in a smaller number of hidden layers and units. The number of units in the selection layer is K_{rsd}. For the presented experiments, a dropout rate of 0.2 was selected. We train RSD for 800 epochs with $\tau_e = 0.1$ and $\tau_0 = 10.0$. For training, we utilize the Adam solver, with a batch size of 512 at a fixed learning rate of 10^{-3}, and an early stopping if the validation loss does not improve for 80 epochs. The parameters, $\boldsymbol{\Theta}$ and $\boldsymbol{\phi}$, are initialized with a Glorot normal initializer [117]. The dataset is split into 0.8 and 0.2 for training and hold-out

testing, respectively.

To derive the final location estimate, the obtained $\widehat{\mathbf{h}}_r$ from RSD is fed into the subsequent localization and uncertainty aware DNN (LUD), which we discussed in the first part of this chapter. Given $\widehat{\mathbf{h}}_r$, the second network is parameterized by $\boldsymbol{\Psi}$ and defined as $f_{\boldsymbol{\Psi}}^{(\mathrm{LUD})} : \mathbb{R}^{K_{\mathrm{rsd}}} \mapsto \mathbb{R}^4$. Knowing from the first part of the chapter, LUD provides not only estimates of the position but also the uncertainty inherent in the samples and the model parameters $\boldsymbol{\Psi}$. Thus, the final location of the transmitter and its corresponding uncertainty are denoted by $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\boldsymbol{\sigma}}$, respectively.

3.3.5 Localization Performance with Learned RRH

To investigate the selection strategy, we evaluate the performance regarding the localization error for $K_{\mathrm{rsd}} \in \{55, 42, 36, 32, 30\}$. We compare the selection strategy to a low-complexity approach based on maximum channel gain (CG) at each RRH. In this case, we define sort $\max_{K_{\mathrm{rsd}}}(\cdot)$ operation, which provides K_{rsd} indices of RRHs matching to the largest channel gain values in descending order, i.e.,

$$\{z'_1, ..., z'_{K_{\mathrm{rsd}}}\} = \underset{\substack{K_{\mathrm{rsd}} \\ m \in \mathcal{M}}}{\mathrm{sort}\,\max} \sum_r \|h_{r,m}\|^2 . \tag{3.23}$$

Then, the CU selects $\widehat{h}_{r,k} = \widetilde{h}_{r,z'_k}, \forall k$. In Fig. 3.12, we show the averaged RMSE over multiple realizations of LUD and provide 95% confidence interval for different $|\mathcal{K}_{\mathrm{rsd}}|$. We see that the proposed selection strategy allows us to learn a subset of RRHs at a price of a small positioning performance loss. Compared to the straightforward CG approach, our method has a significant gain, especially when $K_{\mathrm{rsd}} \ll M$.

To possibly understand how the selection pattern change with K_{rsd}, next we investigated the probability of selecting $m \in \mathcal{M}$ as we vary the number of selected RRHs $|\mathcal{K}_{\mathrm{rsd}}|$.

Relevance of RRH Positions

An additional advantage of using the RRH selection strategy is that it allows us to identify and possibly interpret the most relevant positions of the RRHs. This can be beneficial, especially when it is hard to predict the propagation environment a priori, and thus no optimal configuration design of RRH positions can be foreseen. Fig. 3.13 depicts the $K'_{\mathrm{rsd}} = 15$ most frequently selected RRHs among $K_{\mathrm{rsd}} \in \{55, 42, 36, 32, 30\}$ and $\gamma = 3.0$. It can be noticed that the learning strategy consistently selects a number of RRHs and favours those in corners of the ROI. This suggests that certain positions of spatially distributed RRHs provide a more diverse view of the propagation environment and thus have a higher impact on minimizing the localization error during the selection process.

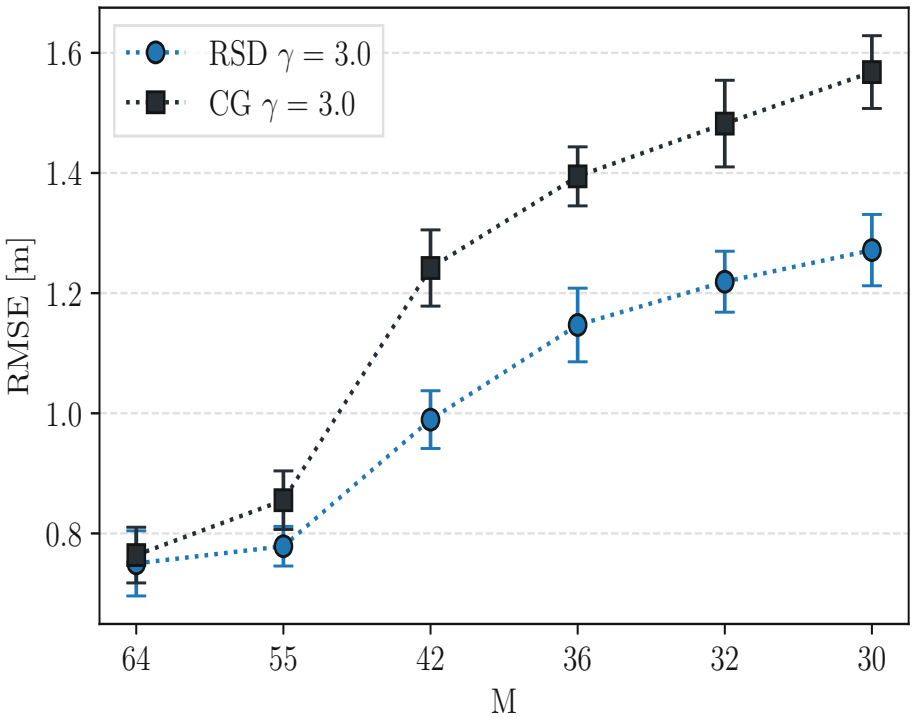
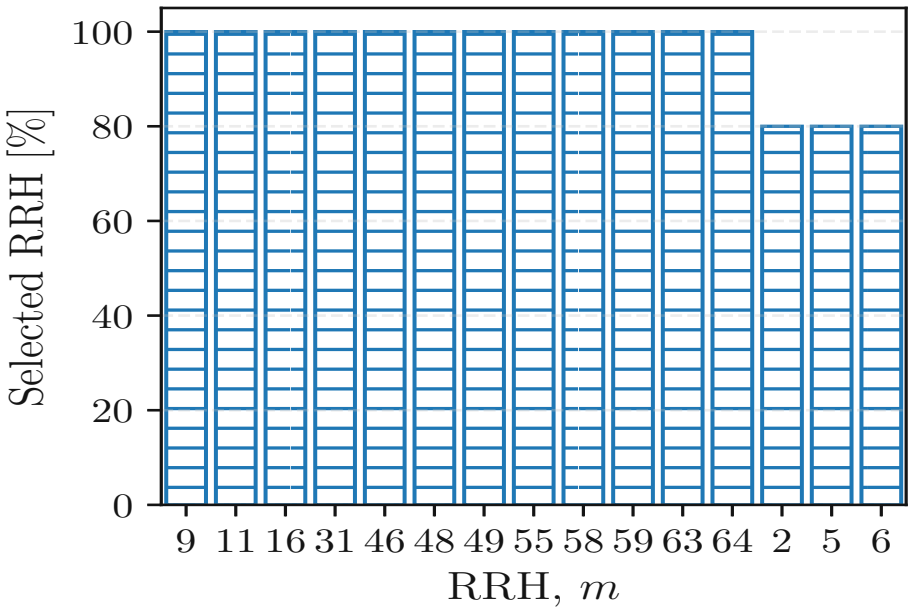


Figure 3.12: Localization accuracy for different M and $\gamma = 3.0$. As the compression ratio increases, the performance gain is higher for RSD.

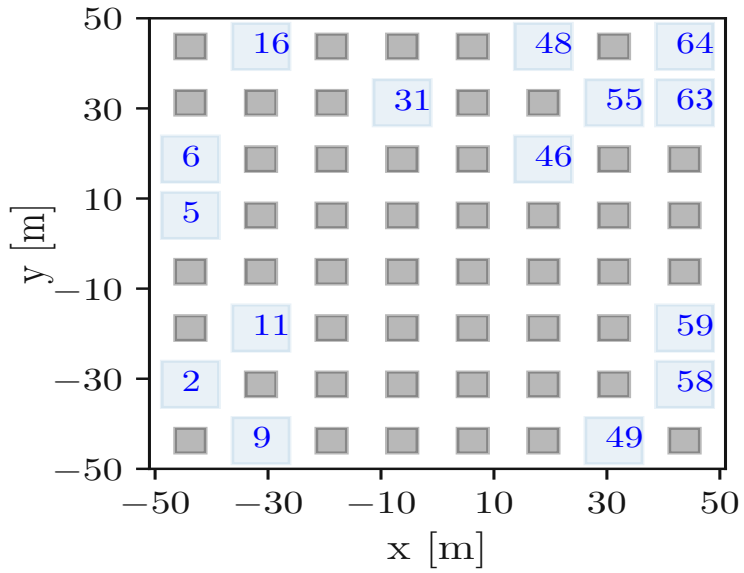
3.4 Final Remarks

In this chapter, we have discussed the challenges of current DNN-based wireless localization techniques. Specifically, we have addressed the limitations of supervised techniques, which neglect to consider the uncertainty stemming from changing propagation conditions and the lack of sufficient training sample size. We introduced LUD, a simple yet effective approach that integrates a stochastic, MC-based sampling approach with a GMM to simultaneously output UE location coordinates, altogether with data and model uncertainty information. Although the measured uncertainty does not provide an exact calibrated measure of variance, we showed that the uncertainty scores of LUD are highly correlated with the actual RMSE values. This is particularly true for large errors and the deep ensemble approach (i.e., LUD with DEN). Hence, the obtained measures can still serve as a heuristic for decision-making and provide relevant insights into the reliability or quality of predictions.

Even though not in the scope of this dissertation, another valuable application of uncertainty measure discussed in this chapter is its role in importance sampling [118]. In ML community, the foremost application of importance sampling is active learning that employs an uncertainty-based sampling query strategy [119, 120]. Selecting only channel estimates that the model is least certain to predict, can not only reduce the



(a) Frequency of selected top $K'_{\text{rsd}} = 15$ RRHs



(b) $K'_{\text{rsd}} = 15$ RRHs

Figure 3.13: The most relevant RRHs in the ROI. a) The occurrence of the most selected RRHs. b) Actual positions of the $K'_{\text{rsd}} = 15$ RRHs.

overall localization error, but also reduce the number of channel estimates required to label for training or fine-tuning the model. We use the uncertainty-based sampling query strategy to demonstrate the impact of training samples on localization error in Appendix B.3.

While in the first half of the chapter we focused on localization with channel estimates acquired at a BS with co-located antennas, in the other half we extended the investigation to DAS. To alleviate the fronhaul capacity constraints, we elaborated a new approach based on Gumbel extreme-value distribution for selecting a subset of RRHs, which allows end-to-end learning over a discrete set of RRHs. We show that the selection strategy comes at the cost of small performance degradation. Finally, the learned parameters in the RRH selection layer can be further probed to allow the analysis of the position importance of the RRHs, possibly helping the interpretation and facilitating design choices.

4

From Shallow to Deep Metric Learning for Localization

One of the primary challenges in wireless localization with supervised ML models, notably with the neural networks we discussed in Chapter 3, is the ability to capture useful discriminatory channel features with limited labelled training data. With the increased number of antennas and limited training examples, the model can easily overfit the correlated channel at different antennas, increasing the risk of capturing irrelevant channel features and making the task of localization harder. Unlike supervised methods, unsupervised DR techniques can be used to extract the most relevant features from the CSI without the need for labelled data. Therefore, even when represented in a lower-dimensional space, the low-dimensional representation of the esimated channel at a massive MIMO BS should preserve, at least approximately, its intrinsic properties. For instance, one can employ a shallow DR technique (e.g., PCA) that orthogonally decomposes the acquired channel, eliminating redundancies and mitigating the channel estimation noise by maximizing the variance of the received signal at the multiple antenna BS. As a result, the obtained low-dimensional features (latent representation) should be sufficient to capture a channel signature necessary for positioning the UE. However, many shallow DR approaches assume that data residing on a low dimensional linear manifold. Such an assumption may be too strict when considering complex multipath propagation environments and the impact of other system imperfections.

In this chapter, we discuss DR approaches to obtain a channel representation useful for positioning UEs. Specifically, we first investigate shallow approaches based

on principle component analysis (PCA) and iterative scaling. Subsequently, we propose a metric-learning method using neural networks. In both cases, we seek to learn channel features reflecting the relative distance between the UEs, given their estimated channel at the BS. The work described in this chapter is published in [83, 84].

4.1 PCA and Iterative Scaling

PCA PCA is the most widely used technique for lossy data compression or feature extraction. Hence, we first illustrate the efficacy of the PCA-derived channel subspace when used to determine the UE location. To do so, let us consider only a single-path and LOS channel. Then, for a single coherence bandwidth and under LOS propagation conditions, the received channel vector from an UE position $\mathbf{u}_r$ acquired at a massive MIMO BS located at $\mathbf{b}_0$ becomes

$$\mathbf{h}_r = (\parallel \mathbf{b}_0 - \mathbf{u}_r \parallel)^{-\rho} \mathbf{\Gamma}(\phi) \in \mathbb{C}^{N_r \times 1}\,, \tag{4.1}$$

where ρ is the path-loss exponent, $\mathbf{\Gamma}(\phi)$ is the array steering vector for an arbitrary array geometry. For simplicity in our analysis, we consider an uniform linear array (ULA). Therefore, we assume that the channel coefficients of $\mathbf{h}_r$ are considered to be dependent only on the distance $\|\mathbf{b}_0 - \mathbf{u}_r\|$ and the direction of arrival ϕ. Consequently, for a static channel between the UE and the BS with fixed transmit power, the covariance of the channel is tightly coupled with the location, i.e., the maximum channel gains can be given by eigenvectors of $\mathbb{E}[\mathbf{h}\mathbf{h}^{\mathrm{H}}]$ corresponding to the largest eigenvalues, where the expectation can be evaluated over the ensemble of different realizations of $\mathbf{h}$ over a specific sub-region or over T different realizations according to the uncertainty model we introduced in Sec. 2.1.3.

In the following, we assume to have a dataset from R distinct UE locations, and construct an $R \times N_r$ matrix,

$$\mathbf{H}_{\mathrm{ref}} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \vdots \\ \mathbf{h}_R^T \end{bmatrix}. \tag{4.2}$$

Our goal is to construct a D-dimensional representation map of locations solely from the high-dimensional CSI samples. The derived map can be thought of as extracting a low-dimensional representation of UEs spatial information from their corresponding high-dimensional channel vectors $\{\mathbf{h}_r\}_{r=1}^R$. The objective of DR is to preserve certain properties of the data (e.g., variance), while transforming a set of high-dimensional features into a new set of lower-dimensionality. In our case, we desire to maintain the *distance* between two locations in $\{\mathbf{u}_r\}_{r=1}^R$ with the intention that it matches the pairwise dissimilarity between the representations of the respective channel vectors

in the D-dimensional map, $\mathcal{Z} = \{\mathbf{z}_r \in \mathbb{R}^D\}_{r=1}^R$. To achieve this, we first compute

$$\bar{\mathbf{R}}_{\mathbf{H}_{\text{ref}}} = \mathbb{E}\{\bar{\mathbf{H}}_{\text{ref}}\bar{\mathbf{H}}_{\text{ref}}^H\}\,, \tag{4.3}$$

where $\bar{\mathbf{H}}_{\text{ref}} = \mathbf{H}_{\text{ref}} - \mathbf{1}\left(\frac{1}{R}\mathbf{H}_{\text{ref}}^T\mathbf{1}\right)^T$ is the zero-mean sample channel matrix, and $\mathbf{1}$ is a column vector of ones. In the following, we consider only the magnitude part, $\mathbf{R}_{\mathbf{H}_{\text{ref}}} = |\bar{\mathbf{R}}_{\mathbf{H}_{\text{ref}}}|$. To find the optimal linear transformation that exploits the pairwise dissimilarity between the channel from R user locations, we can simply maximize the trace of the sample covariance, i.e.,

$$\begin{aligned} \underset{\mathbf{W}}{\text{maximize}} \quad & \text{tr}(\mathbf{W}^T\mathbf{R}_{\mathbf{H}_{\text{ref}}}\mathbf{W}) \\ \text{subject to} \quad & \mathbf{W}^T\mathbf{W} = \mathbf{I}\,. \end{aligned} \tag{4.4}$$

The goal is to find a transformation where the channel vectors corresponding to different user locations are as distinct as possible, which is achieved by maximizing the spread (variance) in the transformed space. The optimal weight matrix $\mathbf{W} \in \mathbb{R}^{R\times D}$ contains the columns that are the eigenvectors of $\mathbf{R}_{\mathbf{H}_{\text{ref}}}$ corresponding to the largest eigenvalues. In our case, the low-rank approximation of the channel from R locations obtained via PCA is actually computed by eigenvalue decomposition on the empirical covariance matrix $\hat{\mathbf{R}}_{\mathbf{H}_{\text{ref}}} = \mathbf{U}_{\text{PCA}}\mathbf{\Sigma}\mathbf{U}_{\text{PCA}}^T$, where $\mathbf{U}_{\text{PCA}}$ are the left singular vectors of $\bar{\mathbf{H}}_{\text{ref}}$ and the eigenvectors of $\hat{\mathbf{R}}_{\mathbf{H}_{\text{ref}}}$. Similarly, $\mathbf{\Sigma}$ are the eigenvalues of $\hat{\mathbf{R}}_{\mathbf{H}_{\text{ref}}}$, i.e., squared singular values of $\bar{\mathbf{H}}_{\text{ref}}$. The low-dimensional map is considered as

$$\mathbf{Z}_{\text{PCA}} = [\sqrt{\sigma}_1\mathbf{u}_{\text{PCA},1}, \ldots, \sqrt{\sigma}_D\mathbf{u}_{\text{PCA},D}], \tag{4.5}$$

matching the first D largest eigenvalues from $\{\sigma_1, \ldots, \sigma_R\}$ with the corresponding eigenvectors $\{\mathbf{u}_{\text{PCA},r}\}_{r=1}^R$.

In principle, we can view the error between the original channel matrix $\mathbf{H}_{\text{ref}}$ and its $D-$rank approximation as $\sum_{i=D+1}^{N_r}\sigma_i$.

MDS As an optimization problem, PCA is closely related to the so called MDS [121], or metric MDS. MDS is often considered a manifold-learning method to obtain a representation *map* based on the dissimilarities between the points. This is particularly appropriate when such dissimilarities can, exactly or approximately, be expressed in terms of the Euclidean distances. Hence, let us consider a distance matrix $\mathbf{C}_{\text{dis}}$, whose elements

$$d_{i,j}^2 = \|\mathbf{h}_i - \mathbf{h}_j\|^2 = (\mathbf{h}_i - \mathbf{h}_j)(\mathbf{h}_i - \mathbf{h}_j)^H \tag{4.6}$$

are the squared *distances* between the row channel vectors of $\mathbf{H}_{\text{ref}}$ corresponding to two UE locations. The distance metric must satisfy the properties of non-negativity, identity, symmetry, and the triangle inequality. Then, the classical scaling cost

function is

$$\mathcal{L}(\mathbf{Z}) = \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} \left(d_{i,j}^2 - \|\mathbf{z}_{\text{MDS},i} - \mathbf{z}_{\text{MDS},j}\|^2 \right) . \tag{4.7}$$

Its minimum, once again, can be obtained by the eigendecomposition of $\mathbf{B}_{\text{dis}} \approxeq \bar{\mathbf{H}}_{\text{ref}} \bar{\mathbf{H}}_{\text{ref}}^H$, the elements of which can be evaluated by double-centering $\mathbf{C}_{\text{dis}}$, i.e., by computing

$$\mathbf{B}_{\text{dis}} = -\frac{1}{2} \mathbf{J} \mathbf{C}_{\text{dis}} \mathbf{J} , \tag{4.8}$$

where

$$\mathbf{J} = \mathbf{I} - \frac{1}{R} \mathbf{1}\mathbf{1}^\top . \tag{4.9}$$

Therefore, classical scaling, through the process of eigendecomposition and minimizing the cost function that preserves pairwise distances, implicitly maximizes the variance in the reduced-dimensional space. This principle of variance maximization aligns classical scaling with PCA, as in both cases, we seek to represent data in a way that captures the most significant variance along specific dimensions, which is a widely recognized relationship between the two [122, 123].

MDS with Sammon mapping In addition to using PCA to construct a low-dimensional map representation by optimizing a convex objective using eigendecomposition, we next look into an alternative to classical scaling, which is known as Sammon mapping [124]. Sammon mapping is an iterative, gradient-based, approach to evaluate a non-convex objective function and is considered a generalization of metric MDS. In this case, we seek to find an optimal representation by normalizing the squared-errors using the pairwise distance in the original features space. Hence, we adapt the cost function defined in (4.7) such that it subsequently assigns weights to each pair of channel vectors, with the weights being proportional to the inverse of their pairwise distance $d_{i,j}$,

$$\mathcal{L}(\mathbf{Z}_{\text{MDS}}) = \sum_{i=1}^{R-1} \sum_{j=i+1}^{R} \frac{\left(d_{i,j}^2 - \|\mathbf{z}_{\text{MDS},i} - \mathbf{z}_{\text{MDS},j}\|^2 \right)}{d_{i,j}^2} . \tag{4.10}$$

While PCA tends to preserves inter-distances for larger dissimilarities, Sammon mapping gives a greater degree of importance to inter-distances for smaller ones.

4.1.1 Location Estimation using D-dim Features

The essential idea of localization using DR techniques to construct a low-dimensional reference map is depicted in Fig. 4.1, when $D = 2$. For each location $\{\mathbf{u}_r \in \mathbb{R}^2\}_{r=1}^R$,

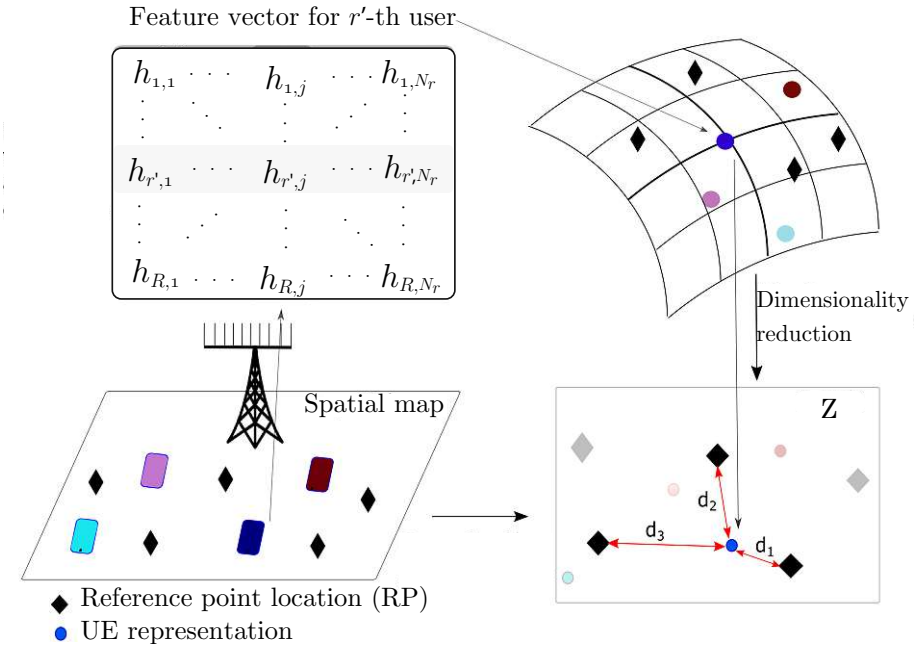


Figure 4.1: Overview of UE location estimation from high-dimensional CSI using dimensionality reduction techniques.

the low-dimensional reference map (i.e., Z-map in Fig. 4.1) has a corresponding feature vector $\{\mathbf{z}_r \in \mathbb{R}^D\}_{r=1}^R$. These feature vectors can be interpreted as pseudo locations. However, there is no unique representation of the data in the Z-map that can gave rise to the actual locations. However, the mapping from locations to channel vectors is not a bijection; i.e., in a general multipath scenario, there are many possibilities for creating the same channel vectors from different locations and propagation environments. Hence, in general, we cannot have a unique backwards mapping. Furthermore, if we know the reduced feature vectors, this tells us *nothing* about the UEs actual values of location coordinates. Therefore, for the purpose of localization, we consider that a subset of CSI obtained at the BS is from reference points (RPs), for which we know the true values of location coordinates. During the localization phase, the constructed low-dimensional features $\mathbf{z}_{r'}$, from the collected CSI at the unknown UE location are matched to those of RPs. As a matching algorithm, we can apply the nearest neighbour search or triangulation to estimate the final position of the transmitter. Given a new UE appears in the Z-map, we once again form the distance metric for each $r \in \mathcal{R}_{\text{rps}}$ as

$$d_{\mathbf{z}_r} = \|\mathbf{z}_{r'} - \mathbf{z}_r\|\,, \tag{4.11}$$

where $\mathcal{R}_{\text{rps}}$ is a set of RP locations. Then, considering nearest neighbour localization, we choose $r_{NN} = \arg\min_{r\in\mathcal{R}_{\text{rps}}} d_{\mathbf{z}_r}$. Finally, the known location with minimum

distance is selected as the estimated position of the UE, $\hat{\mathbf{u}}_{r'} = \mathbf{u}_{r_{NN}}$.

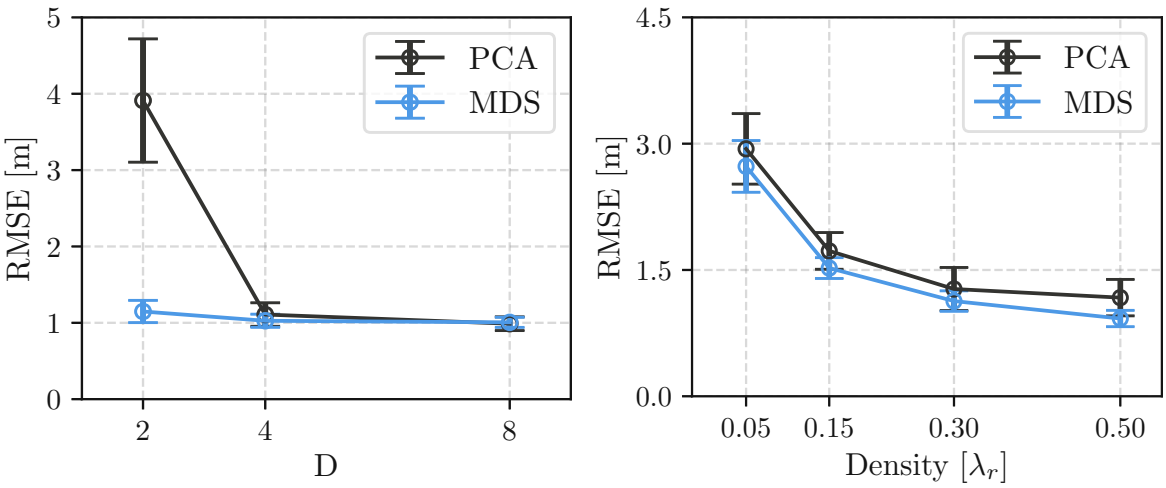
Acquiring the reference map poses a challenge in determining the optimal number of low-dimensional features, D. In terms of minimum error formulation, using $D = \text{rank}(\mathbf{H}_{\text{ref}})$ would naturally result in a reconstruction error of zero, i.e., maintaining all principal components and preserving the variance of the original data matrix $\mathbf{H}_{\text{ref}}$. However, this does not necessarily ensure a minimized localization error. Furthermore, our objective is to attain $D \ll N_r$. A straightforward approach to select D is to quantify the fraction of variance explained by the selected eigenvalues in $\bar{\mathbf{H}}_{\text{ref}}$, i.e., $\sum_{i=1}^{D} \sigma_i / \sum_{i'=1}^{R} \sigma_{i'}$. Alternatively, we can directly evaluate the localization error while sweeping over the value of D. To illustrate the capability of classical DR techniques in retaining low-dimensional features useful for the localization task, in the following, we describe a numerical experiment to evaluate the impact of D.

We set up a simple two-dimensional ROI with a layout of 40m $\times$ 40m and assume that the user locations are distributed based on a Poisson point process (PPP) with the density λ_r. Hence, the probability of having R users in the plane is $\mathbb{P}(R, \mathcal{A}) = [\lambda_r S(\mathcal{A})]^r e^{-\lambda_r S(\mathcal{A})}/R!$ [125], where $S(\mathcal{A})$ is the area of the bounded ROI, $\mathcal{A}$. We place the BS at the origin $(0, 0)$ with respect to $y-$axis. The BS employs a large ULA with $N_r = 64$ equally spaced omnidirectional antennas. The carrier frequency considered is 3.5 GHz, the pathloss exponent is $\rho = 2$, and the SNR is set to 30 dB to account for a channel estimation error. We assume that there are $\lambda_r = \{0.05, 0.15, 0.30, 0.50\}$ users/m^2 and consider 100 randomly selected test user locations for evaluating the localization error.

In Fig.4.2a, we observe that the performance of PCA improves significantly when increasing D from 2 to 4, but then for larger D it saturates. On the other hand, the iterative MDS with Sammon mapping approach, is more robust against the changes in the dimensionality, maintaining a similar performance irrespective of the value of D. Yet, the accuracy hardly improves for $D > 4$. We also investigate the localization accuracy while varying the density of RPs and keeping $D = 4$. As one can expect, and as shown in Fig.4.2b, increasing the density of known locations improves the localization performance for both PCA and MDS.

4.2 Deep Metric Learning

In the first part of this chapter we discussed shallow metric learning techniques. We started with PCA as a spectral method. We then illustrated the performance of an iterative, manifold learning, approach through the use of Sammon mapping cost function. In this part, we continue the discussion on neural networks in general and deep metric learning in particular. In contrast to shallow classical metric-learning approaches, in deep metric learning, we obtain an embedding space using neural



(a) The impact of D.

(b) Density of RPs.

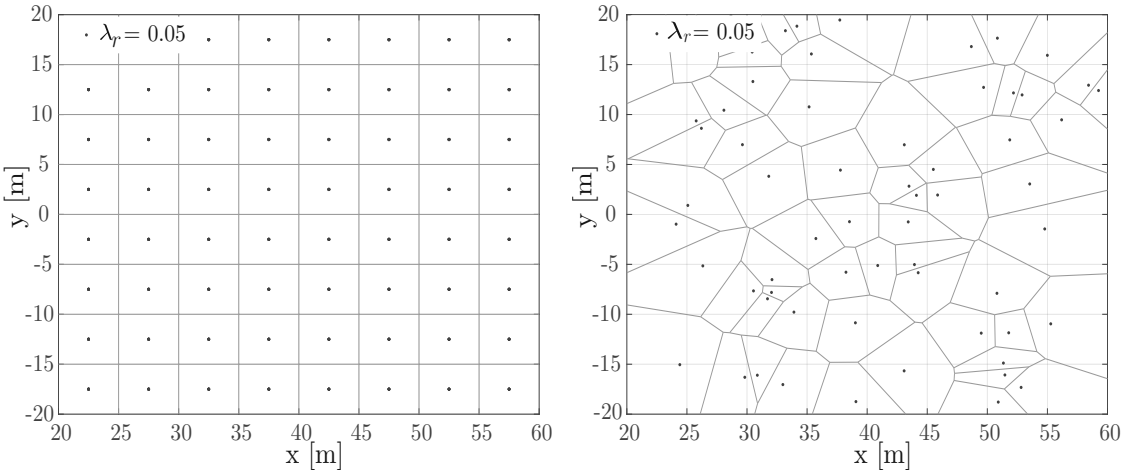
Figure 4.2: Shallow DR techniques for wireless localization. In (a), the impact of D is evaluated while $\lambda_r = 0.5$, and (b) shows the performance accuracy with increasing density of reference locations when $D = 4$.

networks. Hence, the objective of the DNN becomes to learn a D-dimensional embedding that discriminates dissimilar point clouds while simultaneously bringing similar ones closer together. The DNN can map the estimated channel into a metric-defined subspace. Within this space, any distance metric $d(.,.) : \mathbb{R}^D \to \mathbb{R}$ can be applied to match the derived embedding with the closest RP. Furthermore, we can also incorporate a distance metric-based loss function directly in the embedding space.

A commonly used DNN architecture for channel subspace representation learning is the AE [61] (see Sec. 1.2.2). The AE approach consists of encoder and decoder DNNs, separated by a *bottleneck* layer in the middle of the two. For instance, if we consider the real-valued channel $\mathbf{h}_r \in \mathbb{R}^{2N_r}$ as input to the encoder, then the encoder transforms it to an estimate of a $D-$dimensional embedding in the *bottleneck* layer, and the decoder, transforms the embedding back to the input channel, $\hat{\mathbf{h}}_r \in \mathbb{R}^{2N_r}$. Let us define the estimated reconstruction of input channel as $\hat{\mathbf{h}}_r = f_{\mathbf{\Psi}}^{(\mathrm{AE})}(\mathbf{h}_r)$, where $\mathbf{\Psi}$ are the parameters of the AE-based DNN, i.e., $f_{\mathbf{\Psi}}^{(\mathrm{AE})}(\cdot)$. Moreover, if the reconstruction objective is given by the MSE, the squared Euclidean distance,

$$\arg\min_{\mathbf{\Psi}} \mathbb{E}\left[\left\|\mathbf{h}_r - f_{\mathbf{\Psi}}^{(\mathrm{AE})}(\mathbf{h}_r)\right\|^2\right], \tag{4.12}$$

and $D \ll 2N_r$ (also known as under-complete AE [126]), then such an AE is the most illustrative unsupervised deep learning method for creating a metric-based embedding



(a) RPs in squares. (b) RPs random.

Figure 4.3: Example of ROI $S(\mathcal{A})$ partitioning schemes. (a) shows a *grid scenario* with $\lambda_r = 0.05$, and (b) shows the *random scenario* with $\lambda_r = 0.05$.

space. In general, any hierarchical feature extractor (e.g., a multiple-layer CNN) that outputs a latent representation with $D \ll 2N_r$ can be viewed as a non-linear dimensionality reduction method and a metric-learning approach if trained on a metric.

Among the most recognized deep metric learning methods are those based on the Siamese networks [70]. Such methods have been proposed as feature extractors across different domains, such as feature extractors for images of faces [69]. DNN methods, founded on Siamese architecture, consist of multiple equivalent networks sharing identical weights. In this chapter, we introduce an unsupervised, Siamese-based network for obtaining low-dimensional representations useful for localization, detailed further in Sec. 4.2.1. However, prior to investigating that, we first formulate a similar yet supervised approach.

In the following, we consider the multipath channel model introduced in Sec. (2.2). Before feeding CSI into the DNN, we explicitly calculate the average per antenna over all subcarriers. Hence, the input channel vector is $\bar{\mathbf{h}}_r \in \mathbb{R}^{2N_r}$. In this context, we consider a classifier that can learn to separate R_{rps} locations from which we obtain the channel by employing the cross-entropy loss,

$$\mathcal{L}(\mathbf{y}, \bar{\mathbf{h}}, \boldsymbol{\Psi}_b) = \arg\min_{\boldsymbol{\Psi}_b} \mathbb{E}\left[-\sum_{i=1}^{R_{\text{rps}}} \mathbf{y}_i \log f^{(\text{Base})}_{\boldsymbol{\Psi}_b}(\bar{\mathbf{h}}_i) \right], \tag{4.13}$$

estimated over the batch of training examples, and each $i \in \{1, \ldots, R_{\text{rps}}\}$ corresponds to a sub-region with multiple channel realizations, each with varying locations within

the respective sub-region. In (4.13), $f_{\mathbf{\Psi}_b}^{(\text{Base})}$ is the network architecture of the base DNN we introduced in Chapter 3, now modified to behave as a classifier rather than a regressor. More specifically, instead of an identity layer function at the last layer (i.e., a linear activation), we use a `softmax` activation function at the last layer. Hence, the output values $\mathbf{z}_i = f_{\mathbf{\Psi}_b}^{(\text{Base})}(\bar{\mathbf{h}}_i)$ in (4.13) fall within the range of zero to one. These are the normalized probability *scores* corresponding to the predicted RP location. The number of units for the output layer is equivalent to the number of RP locations, R_{rps}. On the other hand, the true labels, corresponding to R_{rps} locations are one-hot encoded, i.e., $\mathbf{y}_i \in \{0,1\}^{R_{\text{rps}}}$. To formulate the UE localization as a classification problem, similar to the first part of this chapter, we assume that R_{rps} locations $\{\mathbf{u}_r \in \mathbb{R}^2\}_{r=1}^{R_{\text{rps}}}$ are known in advance (i.e., RPs) and are distributed according to a PPP with density λ_r in the ROI. Furthermore, we consider two different partitioning schemes for the ROI, $S(\mathcal{A})$:

1) First, we divide the ROI into R_{rps} squares. Each square denotes a region and determines the system's resolution, accuracy, as well as overall computational complexity. We will refer to this partitioning scheme as the *grid scenario*.

2) The *grid scenario* allows us to control the number of samples per location for a more accurate investigation of the performance of our schemes. However, such a partitioning scheme could be impractical. Therefore, in contrast to the first scenario, we consider that RPs are distributed based on PPP, and a Voronoi tessellation is applied to $S(\mathcal{A})$. We refer to this scheme as the *random scenario*.

The partitioning schemes play a crucial role, particularly for the various mining strategies essential to the proposed model, which we detail in Section 4.2.1. We will elaborate more on the motivation in subsequent discussions. An example of the two partitioning schemes is shown in Fig. 4.3 when $\lambda_r = 0.05$. During the training phase, we train the classifier using the collected CSI samples taken from arbitrary positions within the subareas corresponding to R_{rps} sub-regions. Then, the estimated position of the UE during the operation phase, $\hat{\mathbf{u}}_{r'}$, is a weighted mean over the K_{rps} highest predicted probability scores that correspond to known reference point locations,

$$\hat{\mathbf{u}}_{r'} = \frac{\sum_{k=1}^{K_{\text{rps}}} z_{r',k}\mathbf{u}_{r',k}}{\sum_{k=1}^{K_{\text{rps}}} z_{r',k}} \,. \tag{4.14}$$

The value of K_{rps} depends on the overall classification accuracy of the network. We set $c_{\text{rps}} = 2$ because the majority of elements of the vector $\mathbf{z}_{r'}$ are zero. Increasing the value of K_{rps} does not enhance accuracy, thus it is unnecessary to compute the weighted average over all the predicted scores.

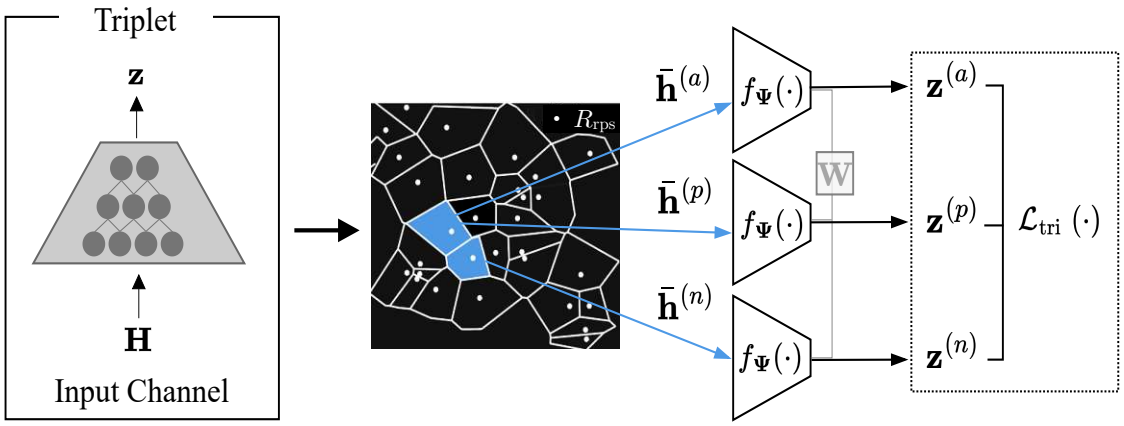


Figure 4.4: Illustration of the triplet-based DNN model.

4.2.1 Triplet DNN and Contrastive Task

In contrast to the conventional formulation of the problem, i.e., the classification task for the base DNN network that we detailed in the previous section, we now discuss the proposed Siamese-based approach to learn a low-dimensional representation of the CSI, depicted in Fig. 4.4. The number of networks used in a Siamese-based method can in principle be any; however, it is usually two or three. A Siamese architecture with three equivalent networks is also called a triplet. Hence, the name triplet network for the model. Specifically, the network is composed of three branches. Each of these networks uses the same hyperparameters as the base DNN for the intermediate computation layers. For the output layer, we use the identity function, $\sigma(\mathbf{z}_r) = \mathbf{z}_r$. Moreover, all three branches share the same parameters. Since the output layer is fixed in terms of the number of units, the numbe of parameters does not increase with increasing the number of RPs. The goal of the *triplet network* is to learn an embedding in the way that the parameters of each network result in the decrease of variance between the channel obtained within the quantized regions and increase otherwise. In other words, our goal is to find an objective function that promotes similarity between the embeddings of two CSI samples acquired from the same location $r \in \{1, \ldots, R_{\text{rps}}\}$ to be more *similar* while pushing apart embeddings corresponding to CSI samples from different regions. Therefore, we sample similar and non-similar CSI; i.e., if two CSI vectors correspond to the same sub-region they are considered similar; otherwise, they are not similar. For obtaining the triplets, we consider $i \in \{1, \ldots, R_{\text{rps}}\}$ as the anchor node, with corresponding $\bar{\mathbf{h}}_i^{(a)}$. Then, we sample one neighboring channel realization belonging to the same region, i.e., a positive sample, $\bar{\mathbf{h}}_i^{(p)}$. Finally, for the negative sample (not similar CSI), we only need to randomly select a channel vector that does not belong to the same sub-region as the anchor node, i.e., the negative sample, denoted by $\bar{\mathbf{h}}_i^{(n)}$. For large-scale scenarios

and/or very high density of RPs, it might not be computationally efficient to sample all the possible triplets. Therefore, one could sample only a portion of the triplets for the sake of the training process. Finally, the obtained triplets for training are given as $\{(\bar{\mathbf{h}}_i^{(a)}, \bar{\mathbf{h}}_i^{(p)}, \bar{\mathbf{h}}_i^{(n)})\}_{i=1}^{R_{\text{tri}}}$, where R_{tri} is the number of triplets providing the triplet network for training.

Since all three networks illustrated in Fig. 4.4 share the same weights, we implement the three branches using a single network, the base DNN. During the training phase, we successively feed the channel realizations within each triplet. Consequently, we obtain their respective embeddings $\mathbf{z}_i^{(a)} = f_{\mathbf{\Psi}}(\bar{\mathbf{h}}_i^{(a)})$, $\mathbf{z}_i^{(p)} = f_{\mathbf{\Psi}}(\bar{\mathbf{h}}_i^{(p)})$, and $\mathbf{z}_i^{(n)} = f_{\mathbf{\Psi}}(\bar{\mathbf{h}}_i^{(n)})$. By collecting all the possible triplets from the ROI under investigation, we then minimize the loss function,

$$\mathcal{L}_{\text{tri}}(\mathbf{z}_i^{(a)}, \mathbf{z}_i^{(p)}, \mathbf{z}_i^{(n)}, \mathbf{\Psi}) = \arg\min_{\mathbf{\Psi}} \sum_{i=1}^{R_{\text{tri}}} \left[\|\mathbf{z}_i^{(a)} - \mathbf{z}_i^{(p)}\|^2 - \|\mathbf{z}_i^{(a)} - \mathbf{z}_i^{(n)}\|^2 + \delta_{\text{tri}} \right]_+ , \quad (4.15)$$

where δ_{tri} enforces a margin between the similar and non-similar pairs, and $[.]_+ = \max(0, .)$. Intuitively, incorporating the negative samples in the triplet-loss function in (4.15) helps the model better distinguish channel estimates between different subregions. This triplet margin, δ_{tri}, is tuned during the training process, and for most of the experiments presented further below, $\delta_{\text{tri}} = 0.2$ yields the best results. In contrast to the direct-classification approach, for the purpose of localization with the triplet network, we need to store the derived embeddings in addition to the coordinates of the ground-truth locations in an embedding collector. Storing the obtained embeddings from the training data is necessary for the direct *comparison* of these embeddings with those from the test data, resulting in the final sub-region classification of the channel. This can be observed as additional overhead for the storage and computation parts of the system. However, this can be considered feasible for many modern systems, as well as applications. During the testing phase, the channel realization of a new UE is passed through the network to obtain an embedding, $\mathbf{z}_{r'}$. In this case, given that a new UE appears in the ROI, same as in the first part of this chapter, we evaluate $d_{\mathbf{z}_r} = \|\mathbf{z}_{r'} - \mathbf{z}_r\|$. Then, we choose $r_{NN} = \arg\min_{r \in \mathcal{R}_{\text{rps}}} d_{\mathbf{z}_r}$. Finally, the location of the RP corresponding to the most similar embedding is retrieved and considered as the position of the transmitter, $\hat{\mathbf{u}}'_r = \mathbf{u}_{r_{NN}}$.

4.2.2 Localization Performance

We evaluate and compare the performance of the triplet network due to the various factors that influence its ability for position estimation of the UE. First, we investigate the positioning performance for two different propagation cases, i.e., when the LOS path is present or absent. We also vary the density of RPs. Afterwards, we study the ability of the model to work with different distribution scenarios of the RPs corresponding to the partitioning schemes of Fig. 4.3. Finally, we look into the

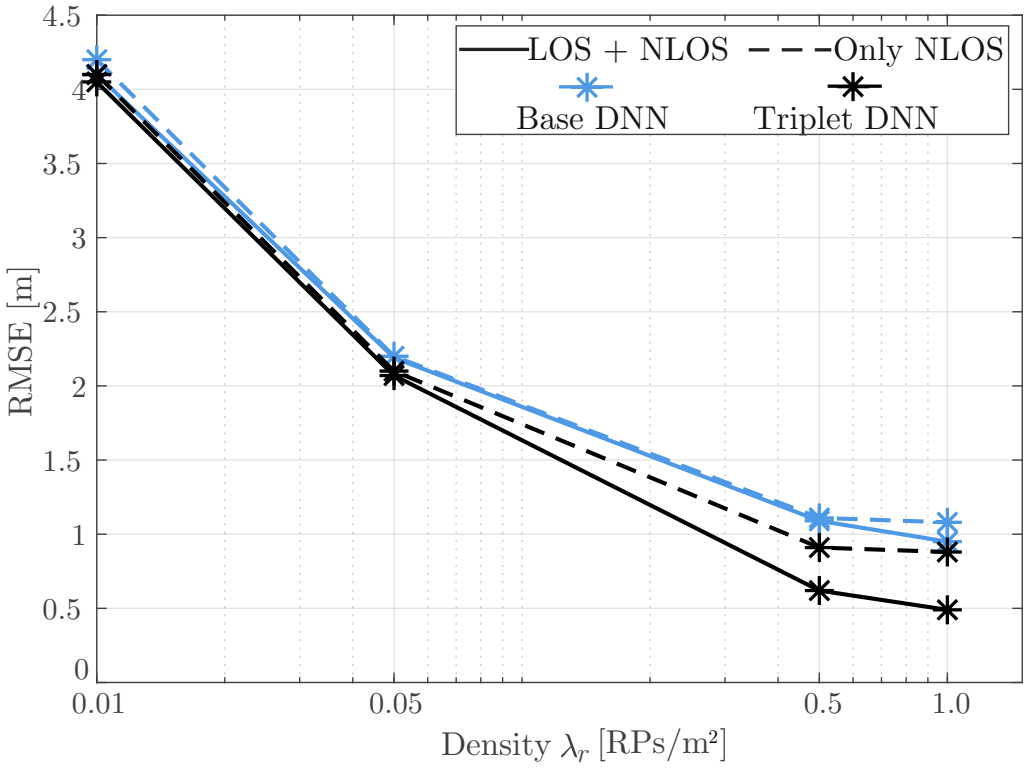


Figure 4.5: The influence of line of sight path and the density of reference point locations for the base DNN and triplet network architecture.

amount of samples required for training the triplet DNN in contrast to the base one.

For the results obtained via the simulations, the model with the lowest validation loss from 100 epochs is selected. Furthermore, when LOS is absent, the triplet loss margin during training is changed to $\delta_{\text{tri}} = 10$. The dataset is split into 80% and 0.2% for training and testing, respectively. The scenario considered is a 40m $\times$ 40m area, the BS at the origin $(0, 0)$, and users distributed in the far field at a minimum distance of 20m from the BS. We consider three single-bounce reflected paths, i.e., $L_{\text{path}} = 3$ NLOS paths, and we set the number of subcarriers to $N_{\text{c}}' = 512$.

LOS and Density of RPs

The density of reference point locations is the parameter that determines the desired positioning accuracy. With the increase of RPs, the system's ability for higher accuracy increases, too. However, the presence or the absence of a LOS path in a multipath channel influences the accuracy of the prediction as well. Consequently, it limits the accuracy even when a high density of RPs is used. For instance, the averaging over subcarriers that we apply may result in a loss of fine-grained

resolution for distinguishing between closely separated locations. Additionally, the high correlation of channel responses at nearby locations could lead to challenges in differentiating these locations, limiting the achievable accuracy, especially when the strong LOS component is absent.

In Fig. 4.5, we show the impact of LOS with the increased density of RPs for both DNNs, the classifier and the triplet network. When the LOS path is present, the increase in density of RPs increases the overall estimation performance. Furthermore, the triplet network outperforms the base DNN classifier when $\lambda_r > 0.05$. However, when the LOS path is absent, the overall accuracy drops for a higher density of RPs compared to the case when the LOS path is present.

As we already noted, the drop in accuracy for denser RPs can happen due to the pre-processing step we apply, i.e., the averaging over subcarriers. In a dense RP setup, two nearby locations might exhibit *identical* large-scale parameters, resulting in highly correlated channels. This can diminish the potential gains in accuracy. Therefore, more carefully designed mining strategies may be required for denser RP sampling in case of the triplet network.

Partitioning Schemes

Collecting ground truth under the assumption of a grid scenario, where each square is a quantized location (a sub-region) and corresponds to one RP, is not very practical. Thus, in Fig. 4.6, we show that both DNNs, without any hyperparameter changes, can be trained and provide similar performance when a random scenario is considered for the investigation of the localization accuracy. We can observe that when the LOS is present and the low density of RPs, both models can provide an error of less than 10m for *grid*, as well as *random scenario*. In contrast, increasing the density of RPs to $\lambda_r = 1.0$, the triplet-based model outperforms the base DNN classifier in both random and grid scenarios.

Amount of Training Samples

In Fig. 4.7, we show the simulation results for the number of samples required for training the base DNN classifier and the triplet-based model. We consider that the ROI is divided into 1 600 sub-regions. For the grid scenario, this corresponds to one m^2 per sub-region for the scenario of $40m \times 40m$, which was illustrated in Fig. 4.3. As we can observe, the triplet network, can provide satisfactory positioning accuracy even with a relatively small amount of labelled data, i.e., 3 samples /m^2 corresponding to a RP. The triplet network outperforms the classifier at the 80th percentile even when the latter DNN is trained with almost 10 times the number of samples. This demonstrates one of the most potent characteristics of metric-learning-based DNN models in general and Siamese-based neural network architectures in particular.

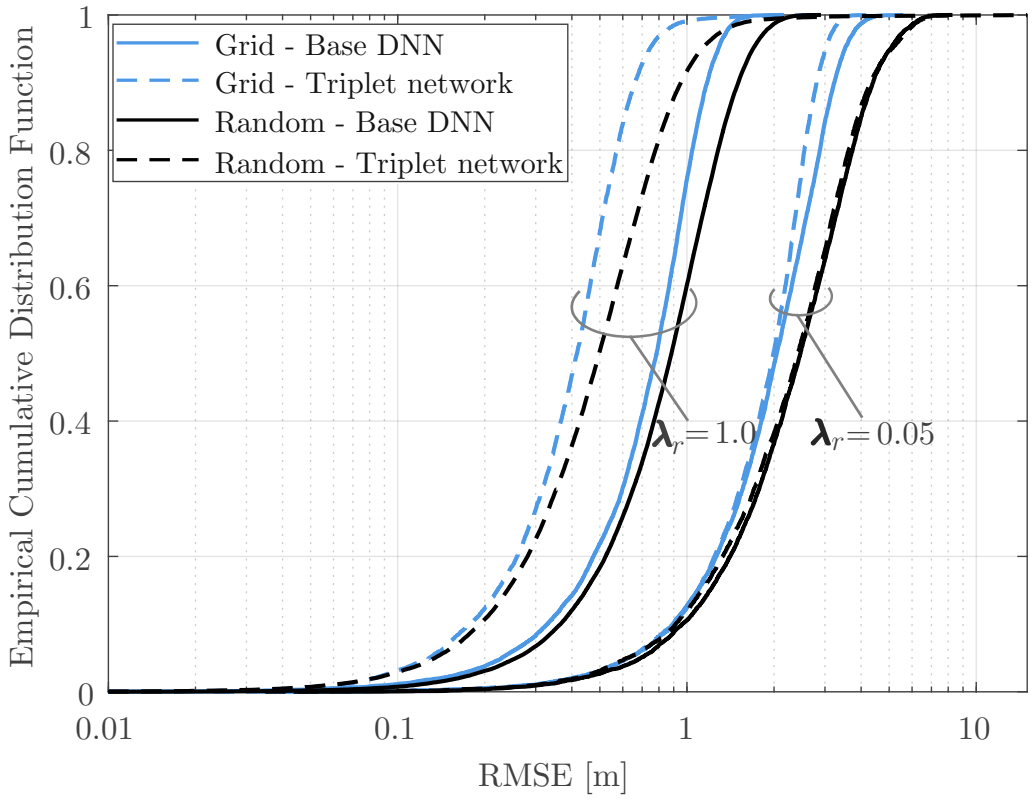


Figure 4.6: Triplet network outperforms the base DNN classifier for a high density of RPs in *grid* and *Voronoi scenarios*.

In the forthcoming chapter, we discuss the disadvantages of the triplet network proposed in this chapter and extend our investigation to a new approach that may hold the promise of a high-accuracy and practical localization system suitable for non-static propagation environments typical for wireless communication systems.

4.3 Final Remarks

Developing a machine learning algorithm that allows us to train with as few ground-truth labels as possible is essential for maintaining high availability. The CSI from the reference point locations in a dynamic scenario can quickly become outdated. Therefore, it is necessary to develop the capability to train the model at smaller time scales, taking into account the temporal characteristics of the environment. In this chapter, we discussed localization schemes based on subspace learning and dimensionality reduction techniques applied to high-dimensional channel estimates available at a massive MIMO BS. Our focus was on the metric-learning formulation of the DR techniques. Our primary goal was to derive a distance metric or embedding

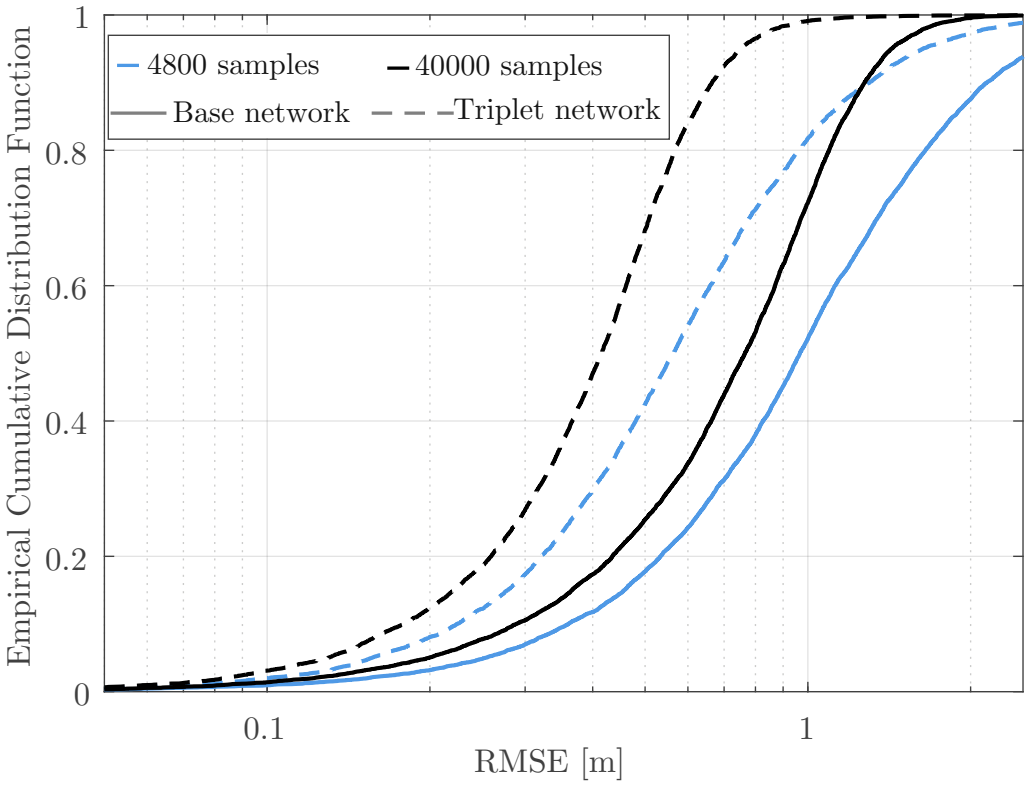


Figure 4.7: Localization accuracy as a function of the amount of training samples.

space to discriminate channels across different locations while grouping together those from similar ones. Initially, we illustrated shallow linear subspace and manifold learning approaches, and then we introduced a Siamese-based DNN model (i.e., the triplet network) for channel dimensionality reduction and feature extraction in the second part of the chapter.

We showed that the triplet network can reduce the high-dimensionality of the CSI whilst maintaining relative proximity with K_{rps} nearest locations depending on the density of RPs. Using the neural network and incorporating a distance metric in the embedding space, we can learn a $D = 4$ dimensional representation of the channel, which is sufficient to capture the relevant channel characteristics to reveal a unique spatial signature of the UE. We showed that the proposed scheme could provide higher accuracy with fewer training samples when compared to the base DNN classifier.

In this chapter, we have employed the Euclidean distance as a measure of dissimilarity, and have assumed partitioning schemes are in place in order to construct both the contrastive task and the triplet-loss function. Overall, to achieve high localization accuracy with a reasonable amount of computational memory, discriminating channels

based on the pairwise distance loss can be challenging. The triplet DNN requires a careful partitioning scheme, a good triplet mining strategy, and assumes a consistent variance between the anchor node and positive samples across different quantized regions corresponding to RPs.

5

Wireless Transformer and Self-Supervised Representations

The ML methods discussed throughout the dissertation operate directly on the measurement data, i.e., the raw channel estimates are the input of a DNN model. For the base DNN classifier discussed in the previous chapter, learning representations useful for wireless localization often necessitate vast amounts of annotated CSI. On the other hand, using the proposed triplet network requires mining triplet pairs for the predefined contrastive task and prior knowledge of the partitioning schemes of the scenarios. Furthermore, both the classifier and the triplet network utilize simple feedforward MLPs. Albeit such neural network components have a straightforward implementation and arguably computational efficiency, they lack the spatial invariance properties, among other disadvantages more specific to wireless communication signal design, that we discussed in Chapter 1. As a result, MLPs can be, for instance, overly sensitive to small-scale channel variations. Hence, in this chapter, we introduce advanced DNN architectures, and aim at addressing the limitations of prior works. We first present a transformer-based method for wireless localization, and subsequently discuss how we extend the supervised model to a self-supervised learning (SSL) approach. SSL, in general, aims to reduce the extensive costs of labeled CSI acquisition and to avoid the additional signal-processing overhead associated with the hand-designed feature extractors or the mining of contrastive pairs. Our goal in this chapter is to learn compressed channel representations beneficial for wireless localization and beyond. More specifically, we strive towards general-purpose channel features invariant to fading and system impairments, that can be transferred to new environments and are ready to use for different wireless network tasks.

5.1 Wireless Transformer

CNNs have been widely adopted for wireless signals, establishing them as state-of-the-art neural network blocks of DNN-based methods for wireless localization. Nonetheless, a fundamental limitation of such CNN-based models is the assumption of inductive bias on the local spatial structure of the input signal information. This assumption, inherent to their design (apt for image signals), suggests that they may be particularly effective at learning wireless channel features in relatively static environments. In such conditions, stationary regions of the channel can be more predictable, and the presumption of translation equivariance is more likely to hold valid. Enforcing a prior is particularly advantageous when data is scarce, allowing CNNs with fewer parameters to learn efficiently. However, with ample training data, CNN's potential might be underutilized. Moreover, the use of convolutional kernels in CNNs makes them disadvantageous in modeling relationships beyond their designated *receptive field*. Methods inspired by [127] can significantly enhance their *receptive field*, but often accomplish this by fusing (e.g., pooling) local dependencies.

On the other hand, the introduction of the attention mechanism [79] has inspired new neural network components, like transformers [77], removing convolutional and recurrent operations and relying only on the *attention* to model large-scale dependencies between input and output. Due to their ability for parallel computations, transformer-based models are more efficient during training when compared to convolutional and recurrent counterparts. Hence, they are now widely used in signal processing for natural language- and computer vision-related tasks. To illustrate, transformer-based architectures span across various domains, like [78, 128], and they employ the self-attention to compute a *weight* value between pairs of input representations. The attention coefficient reflects the mutual dependency between the inputs, e.g., correlation. For instance, in NLP, such input representations are typically words, or parts of it, and are refereed to as word tokens [77]. Similarly, in computer vision, the inputs are associated with groups of pixels, frequently referred to as image patches [78, 129].

In communication systems based on, e.g., OFDM, the CSI at a massive MIMO BS offers a detailed representation of the channel's response across individual OFDM subcarriers. Hence, we conceptualize subcarriers as input representations to capture channel variations across frequency, time, and antennas. This motivates us to consider the self-attention as a neural network processing technique and propose the wireless transformer (WiT). In the following, we will discuss the main architectural components of the transformer-based model for wireless localization. An outline of WiT and its main building blocks is depicted in Fig. 5.1.

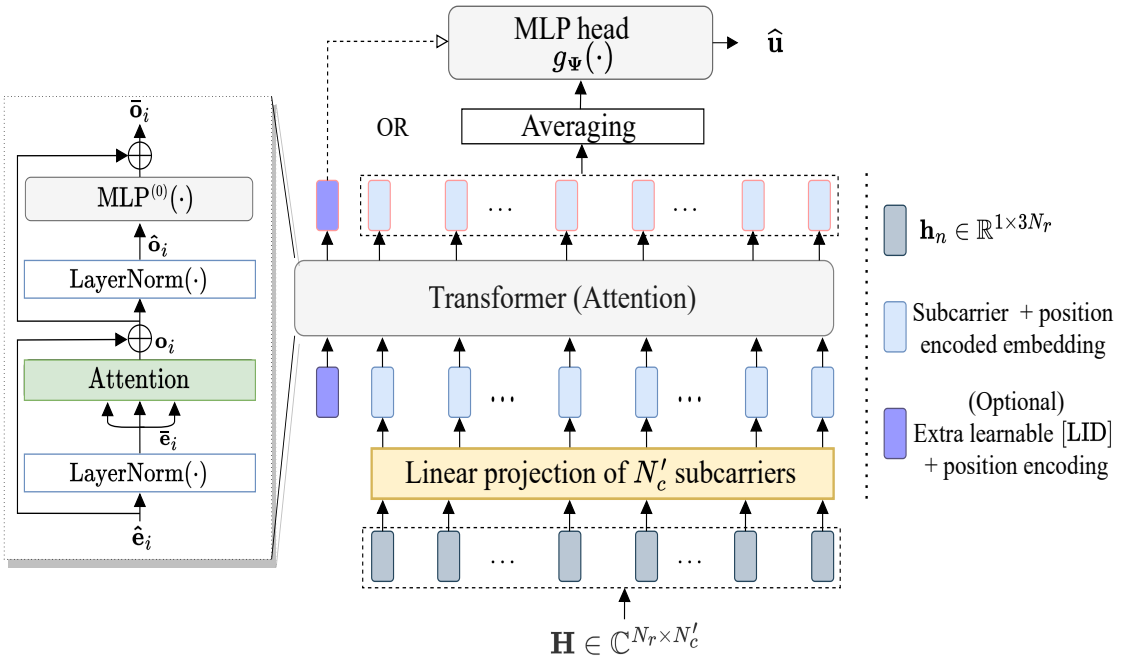


Figure 5.1: Overview of WiT, a transformer-based model. We linearly transform each subcarrier, add positional encoding, and input the channel representations to a transformer block that uses the attention mechanism. For channel-to-location mapping, we average the *attended* features or use the extra learnable symbol, [LID].

5.1.1 Transformer and Attention

First, let us recall a fully-supervised formulation for positioning the UE from the obtained channel estimates. As in previous chapters, we formulate the WiT as a function $f_{\Psi}^{(\mathrm{WiT})}(\cdot)$ parameterized by $\mathbf{\Psi}$ where, given the input channel $\mathbf{H}_r$, we aim to learn a set of robust features and directly map them into position coordinates, $\hat{\mathbf{u}}_r$, i.e.,

$$\mathcal{L}(\mathbf{u}_r, \mathbf{H}_r, \mathbf{\Psi}) = \arg\min_{\mathbf{\Psi}} \mathbb{E}\left[\left\|\mathbf{u}_r - f_{\Psi}^{(\mathrm{WiT})}(\mathbf{H}_r)\right\|^2\right] . \tag{5.1}$$

In the previous chapters, we primarily focused on either selecting a single OFDM subcarrier or averaging over them for inputting to the DNN architecture. However, in this chapter, we change the perspective. We now view the estimated channel matrix $\mathbf{H}_r$ as a set of N_c' channel vectors. In the following, we discuss the primary building blocks of WiT and elaborate on the transformations of input channel representations as they flow through the architecture.

Subcarrier Embedding

As stated several times during this thesis, we handle the complex-valued channel coefficients by stacking their real and imaginary parts. For WiT, as well as other approaches presented in this chapter, we additionally consider the absolute part. Hence, the input representation of each subcarrier (after normalization), shown in Fig. 5.1, becomes $\mathbf{h}_n \in \mathbb{R}^{1\times 3N_r}$. We aim to preserve the per-subcarrier channel structure and exploit the information that each subcarrier conveys, as well as the interdependencies that exist among them. Individual subcarrier representations undergo a linear transformation process. More specifically, we apply a linear transformation to convert subcarriers sequence into linear embeddings, employing a linear layer. This layer has learnable parameters, $\mathbf{E} \in \mathbb{R}^{3N_r\times D}$, leading to the subcarrier embedding $\mathbf{e}_i = \mathbf{h}_i\mathbf{E}$.

Subcarrier Positional Encoding

A characteristic of the transformer in general and the attention mechanism in particular, is the permutation equivariance with regards to the subcarrier embeddedings. Since we use no recurrence or a standard convolutional operator, the model would treat subcarrier embeddings as a set of unordered representations. However, the structure of the whole channel, i.e., the arrangement of the subcarriers, can reveal useful dependencies among frequency-selective subcarriers. Hence, during the training process, we encode positional information in the learning process to make sense of the subcarrier's position in the sequence.

To realize positional encodings, different types of codebooks or pre-defined functions can be used. For instance, [77] proposes to use fixed, non-parametric periodic functions, like sinusoidal patterns, to impart absolute position information. In contrast, learnable positional encodings, for example, enable the model to adjust the position information during the training process. In this case, we initialize a random real-valued vector, denoted as $\mathbf{g}_i \in \mathbb{R}^{1\times D}$, for each subcarrier index i. Then, given the input channel, $\mathbf{g}_i$ is added to the subcarrier embedding $\mathbf{e}_i$ at position i. Hence, the input to the transformer block becomes $\hat{\mathbf{e}}_i = \mathbf{e}_i + \mathbf{g}_i$.

Location Identification

To incorporate context information for the entire channel, we introduce a unique symbol, denoted as [LID]. The [LID] serves a purpose similar to the [CLS] token utilized in the BERT model [130]. This is as another learnable vector, $\mathbf{e}_0$, and its learned representation aims to distill the entire channel's information from $r-$th UE location. Consequently, the total number of input presentations to the transformer block becomes $C_{\text{wit}} = N_c' + 1$.

The output representation $\bar{\mathbf{o}}_0 \in \mathbb{R}^D$, corresponding to the input $\hat{\mathbf{e}}_0$, is then directed to the task-specific MLP, i.e., the MLP head shown in Fig. 5.1. This embedding is relevant for mapping the compressed channel features to the UE location. Furthermore, we can utilize only $\bar{\mathbf{o}}_0$ (i.e., [LID]) for transfer learning to other tasks.

Attention Mechanism

Attention is the main building block of any transformer architecture. There exist different ways for utilizing the attention mechanism. Later in this chapter, we will address variations within a transformer architecture, detailing their roles and computational implications. In case of self-attention, as employed in [77], we consider three input *copies* and project them using the same set of weights, $\mathbf{W}_q = \mathbf{W}_k = \mathbf{W}_v$. Accordingly, we express the self-attention as

$$\mathbf{o}_i = \sum_{j=1}^{C_{\text{wit}}} \frac{\exp\left(\alpha_{i,j}\right)}{\sum_{j'=1}^{C_{\text{wit}}} \exp\left(\alpha_{i,j'}\right)} \left(\bar{\mathbf{e}}_j \mathbf{W}_v\right) , \tag{5.2}$$

where $\alpha_{i,j}$ can be interpreted as the normalized *degree* of dependency between any two embeddings corresponding to positions i and j in the input sequence, i.e.,

$$\alpha_{i,j} = \frac{1}{\sqrt{D}} \left(\bar{\mathbf{e}}_i \mathbf{W}_q\right) \left(\bar{\mathbf{e}}_j \mathbf{W}_k\right)^T . \tag{5.3}$$

Furthermore, the representation $\bar{\mathbf{e}}_i$ is obtained from the layer normalization of $\hat{\mathbf{e}}_i$, $\bar{\mathbf{e}}_i = \text{LayerNorm}(\hat{\mathbf{e}}_i; \zeta, \iota)$ [131]. The hyperparameters ζ and ι stabilize the distribution of activations across layers. Overall, the LayerNorm benefits the training by controlling the mean and variance of individual activations.

The attention mechanism described so far considers a single attention head within the WiT. However, transformer-based architectures can work with multiple attention heads simultaneously, i.e., multi-head attention. Each head has its own set of learnable parameters and operates independently, learning different views of the same input channel. By default, we will maintain the low-computational complexity by utilizing a total number of attention heads $H_{\text{attn}} = 1$ and transformer blocks $H_{\text{blck}} = 1$. Nevertheless, throughout the chapter, we will investigate other configurations as well.

MLP

The subcarrier representation $\bar{\mathbf{o}}_i$ is obtained at the output of the transformer block,

$$\bar{\mathbf{o}}_i = \text{MLP}^{(0)}\left(\hat{\mathbf{o}}_i\right) + \left(\mathbf{o}_i + \hat{\mathbf{e}}_i\right) , \tag{5.4}$$

where $\hat{\mathbf{o}}_i = \text{LayerNorm}\left(\mathbf{o}_i + \hat{\mathbf{e}}_i; \zeta, \iota\right)$, and the $\text{MLP}^{(0)}(\cdot)$ is a two-layer feedforward neural network with D hidden units per layer and non-linear activation functions.

The LayerNorm parameters, ζ and ι, are learnable parameters representing the scale and shift that are applied during normalization. Later in this chapter, we will address further details, including the activation functions.

Additionally, the input to the MLP head (denoted by $\boldsymbol{g}_{\boldsymbol{\Psi}}(\cdot)$ in Fig. 5.1), can be $\bar{\mathbf{o}}_0$ or an averaged embedding over the N_c' representations. In case of $\bar{\mathbf{o}}_0$, for instance, then $\widehat{\mathbf{u}}_r = \boldsymbol{g}_{\boldsymbol{\Psi}}(\bar{\mathbf{o}}_0)$. For the MLP head, we consider either a single linear-layer or a three-layer feedforward neural network, which includes two layers with non-linear activation functions followed by a final channel-to-location (linear) mapping layer.

We explained the main components of WiT, as well as the signal processing flow. When we refer to WiT, we generally refer to the transformer model trained in a fully-supervised setting. In Sec. 5.3, WiT serves as the fundamental building block for the SSL method. However, before elaborating on the new approach, we present the methodology for collecting CSI based on ray-tracing, used for assessing the performance of WiT in a dynamic scenario. In particular, we present two synthetic datasets obtained in two railway scenarios; one relies on a single BS, and the other assumes a distributed massive MIMO setup.

5.2 Ray-Tracing Datasets for Dynamic Scenarios

For assessing the gains of new ML algorithms and tracking the overall research progress, standardized datasets are desired. Although datasets derived from actual measurements are ideal, they are often difficult to obtain across different scenarios or system configurations. Additionally, the wireless channel is time-varying, and often non-stationary. The dynamic nature of the propagation environment means that CSI can quickly become outdated, especially in outdoor vehicular settings. This can occur due to minor changes, such as a vehicle passing by, or longer-term effects, e.g., a new construction site.

Recently, the application of ray-tracing (RT) has become increasingly favored for site-specific channel realizations [44, 110, 132]. The appeal of RT lies mainly in its deterministic approach to modeling the physical interaction of electromagnetic (EM) waves with obstacles in the surroundings to characterize the multipath channel in terms of, e.g., delays, amplitudes, and angles. Consequently, channel realizations generated through the RT tools inherently guarantee spatial consistency and are a straightforward approach for constructing datasets for specific real-world scenarios with varying system configuration parameters.

Scenarios

In our work, we develop two dynamic scenarios and obtain channel realizations for the respective environments. The modeled scenarios are depicted in Fig. 5.2. They represent two railway tracks in Austria, which we refer to as S- and HB-scenario, named based on their locations. The 'S-scenario' is around the Schwechat area, while 'HB' denotes an ROI near the Vienna Central Station (Hauptbahnhof). Advances in computer graphical processing and the availability of varying geographic information technology (GIS) databases have made it relatively feasible and easy to create comprehensive digital models that can well-approximate real-world scenarios. In our case, initial digital assets encompass mostly buildings imported from OpenStreetMap (OSM) [133]. We convert the imported OSM data into polyhedrons, or, more generally speaking, *mesh objects*, using 3D computer aided design (CAD) tools, such as [134]. We use [134] to design other object shapes (e.g., trains, cars, and vegetation) and, more importantly, to simulate dynamic scenarios. To realize a dynamic scenario, we change the position of static objects, e.g., buildings, over T successive realizations (snapshots), according to the uncertainty model that we introduced in Chapter 2, Sec. 2.1.3. In addition, we simulate the *movement* of trains and other vehicles in the surroundings, assuming a constant speed over predefined trajectories. We then export T different scenario description files for feeding into the ray-tracing tools.

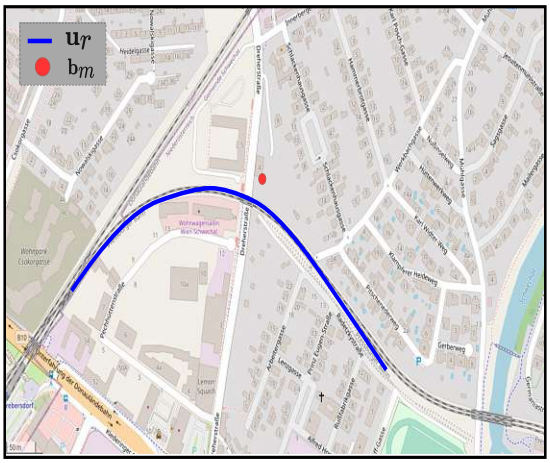
Ray Traces

To obtain all the multipath related parameters for the S- and HB-scenarios, we use the available shooting and bouncing ray (SBR) approach with low-angular separation [135] in the ray-tracing tool from Matlab [136]. We follow a similar approach for other ray-tracing tools, like BJTU-RT [137]. Scenario model files, ray parameters, and generated channel realizations are publicly available [1] from both tools. For the experiments in this dissertation, we use the dataset generated using the Matlab RT. We simulate the temporal variations of the scenarios by running $T = 200$ realizations with altered input geometries, changing the position of the objects in the environment, varying over different material properties, as well as imposing stochasticity in the position of the UE antenna. We consider the default relative permittivity values $\epsilon_{\text{per},\kappa_{\text{per}}}$ for $\kappa_{\text{per}} \in \{\text{concrete, brick, metal, wood}\}$ [96] and add the atmospheric attenuation [97, 138] in the event of rain with $\mathbb{P}(\boldsymbol{\mathcal{R}}) = 0.3$.

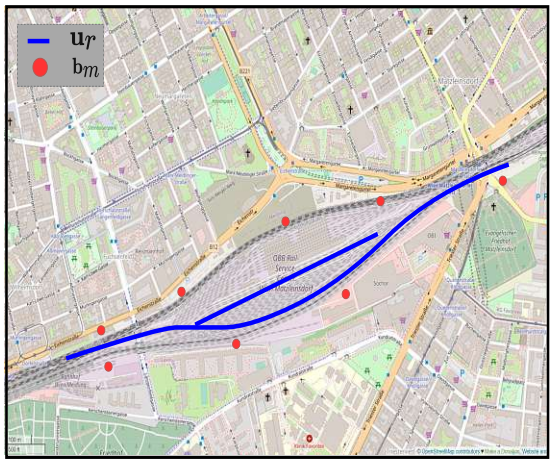
We assume that the UE is situated along the predefined railway trajectory and has an omnidirectional antenna at the initial height of 1.50m. We run RT simulations for R different UE locations over T different snapshots. For the first scenario, S-scenario, we consider a single-BS $M = 1$ and $R = 360$ different UE locations. For the second scenario, HB-scenario, we consider a DAS with $M = 8$ and $R = 406$. RRHs are distributed along both sides of the track, as shown in Fig. 5.2b. We assume $N_r = 64$,

[1] `https://mcg-deep-wrt.netlify.app/deep-wrt/`

$f_c = 3.5\text{GHz}$, $L_{\text{path}} = 4$, and a system bandwidth of $B_{\text{sys}} = 20\text{MHz}$. We consider every 16−th subcarrier as active. The height of RRHs and the BS is set to 20m. The obtained sample size is RT. However, if the received power is less than -130dBm, we discard such measurement at time t from the RT. By this, we obtain datasets of size 69 212 and 81 200 samples for S- and HB-scenario, respectively. In Appendix D.1, we show the RMS delay-spread, τ_{RMS}, as well as the RMS angle of arrival spread in azimuth, φ_{RMS}, for a single random UE location over $T = 200$ time snapshots and $L_{\text{path}} = 4$ strongest paths.



(a) ROI in OSM for S-scenario.



(b) ROI in OSM for HB-scenario.

(c) A 3D rendered digital model example for S-scenario.

Figure 5.2: Railway scenarios for constructing ray-tracing channel measurements.

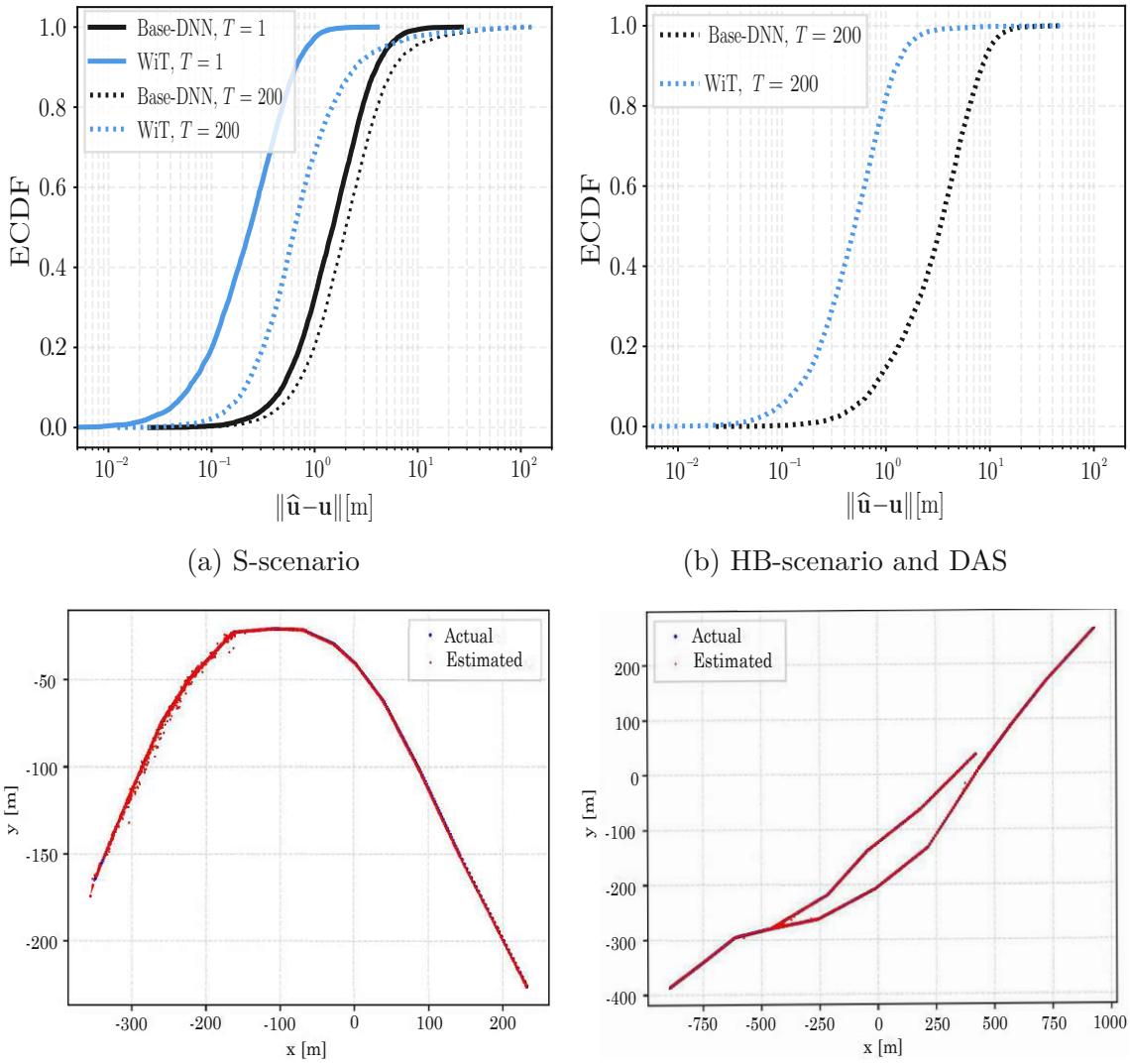
5.2.1 WiT and Dynamic Scenarios

In this section, we assess the performance gains achieved by WiT operating in the above-detailed scenarios. For comparison, we use the base-DNN we discussed in the beginning of Chapter 3. For the sake of consistency, since we are using ReLU and $D = 650$ for the base-DNN, we also keep the same activation function and width for WiT in this evaluation. Specifically, we apply ReLU as a non-linear operation in both MLPs of WiT: $\text{MLP}^{(0)}$ and MLP head. $\text{MLP}^{(0)}$ is composed of two linear layers, with a ReLU operation succeeding each of them. MLP head has two layers as well, where the non-linearity is applied after the first one. Since $u_{r,3} = 1.50\text{m}\ \forall\ r$, we only consider $\hat{\mathbf{u}}_r \in \mathbb{R}^2$. For the LayerNorm$(\cdot)$, the additive factor $\zeta = 1.00$ and the multiplicative parameter $\iota = 0.0001$. We train WiT for 1 800 epochs with a batch size of 512, and we notice that the validation loss does not saturate when trained for longer epochs. On the other hand, we apply a dropout of 0.1 after each layer of the base-DNN and early stopping during the training if the validation loss does not improve for 80 consecutive epochs. The datasets are split into 0.80 and 0.20 for training and validation, respectively. Finally, we use Adam solver with weight decay, and we set the initial learning rate to 3×10^{-4}.

Base-DNN versus WiT

We first investigate the performance of WiT in S-scenario for $T = 1$, i.e. a static environment. To ensure a denser sampling, we reduce the inter-distance between any two UE locations, $\|\mathbf{u}_i - \mathbf{u}_j\|$. Hence, this results in a dataset size of 72 000 distinct labeled channel realizations. In Fig. 5.3a, we show that a substantial improvement in localization accuracy can be achieved when using WiT compared to the base-DNN. The gain can be as much as 50%. Similarly, for the dynamic environment and $T = 200$, the proposed WiT is much more robust compared to using the raw CSI and the base-DNN, reducing the localization error by a significant margin.

Different from S-scenario, in HB-scenario we have N_r distributed antennas among $M = 8$ infrastructure nodes (or RRHs). In Fig. 5.3b, we show the ECDF of the localization error clearly indicates the superior performance of the proposed WiT. It can achieve an improvement with a gap exceeding 7m at the 95−th percentile. In Figs. 5.3c and 5.3d, one can observe the actual versus estimated UE locations on the trajectories under investigation. For the S-scenario, the first part of the track, i.e., the left from the origin, is more prone to prediction errors. This is primarily due to the more challenging propagation, where no dominant or LOS path exists between the BS and UE locations. On the other hand, for the DAS case, the most significant errors occur at the intersection point around the $x = -350\text{m}$ and $y = -270\text{m}$, where the correlation of the estimated channel between very closely spaced UE locations can be high.



(a) S-scenario

(b) HB-scenario and DAS

(c) S-scneario: Actual vs estimated

(d) HB-scneario: Actual vs estimated

Figure 5.3: Localization performance of WiT in a) S-scenario and b) HB-scenario.

Finally, in Table 5.1 we present a comparison highlighting the performance differences between two different approaches of using channel features to feed the MLP head of WiT, i.e., averaging over the representations versus employing only the special symbol [LID]. We show that irrespective of the approach, there is an evident performance gap between WiT and base-DNN in DAS and co-located massive MIMO scenarios. However, it is also notable that an averaged representation outperforms [LID] case.

Table 5.1: Summarized Results

	S $T = 1$		S $T = 200$		HB $T = 200$	
Method	MAE	95−th	MAE	95−th	MAE	95−th
Base-DNN	1.98	5.16	3.59	8.83	4.13	10.01
WiT [LID]	0.74	1.88	2.36	6.54	1.18	2.83
WiT (avg.)	**0.31**	**0.84**	**1.70**	**4.70**	**0.68**	**1.61**

5.3 Self-Supervised Channel Features

In the preceding sections of this chapter, we discussed a transformer-based model (WiT) and investigated its performance in dynamic scenarios. We showed that WiT can exploit the per-subcarrier channel structure and learn more useful channel features than the base-DNN. WiT predominantly relies on learning the large-scale channel characteristics by averaging the learned subcarrier representations or using a unique representation for the entire channel. Nevertheless, the wireless channel experiences both macroscopic and microscopic fading phenomena. Therefore, unlike previous investigations, our objective becomes twofold: we seek to explicitly account for the large-scale variations caused by pathloss and shadowing, as well as the rapidly varying microscopic fading characteristics primarily caused by multipath.

Furthermore, as previously noted, WiT is a fully-supervised technique capable of achieving high localization accuracy. However, supervised learning and transformer-based models are, in general, still limited to scenarios with substantial amount of training data. Despite the abundance of CSI available at the BS, obtaining tagged CSI samples for different wireless network tasks and environments is challenging. Existing DNN approaches typically train and evaluate models separately for each specific scenario, system configuration setting, and task. The transferability and adaptability of pre-trained, fully-supervised models remain unknown. Hence, in the subsequent discussion, we aim to address a fundamental challenge in deep wireless communications, i.e., learning a compressed channel representation that can be used to realize varying downstream tasks, such as wireless localization.

More specifically, for the remainder of this chapter, we extend WiT and propose a SSL method for wireless channel representation learning named SWiT, short for self-supervised wireless transformer. SSL has received significant research attention over the years across numerous fields, including NLP [130], speech recognition [139] and has become among the most promising research areas for computer visual representation learning [140–142]. To learn useful channel features, SWiT depends on solving designed pretext tasks (or auxiliary tasks) using only the available CSI.

In contrast to prior works that rely on metric learning or are based on channel reconstruction (e.g., AE-alike), we aim to leverage the redundant and complementary information across different subcarriers. Hence, unlike the mining of triplets discussed in the previous chapter and forcing to discriminate channels from different sub-regions, we avoid the necessity of sampling negative pairs and employing a contrastive loss.

Building upon the advantages of self-supervised training, SWiT, unconstrained by the availability of any type of labeled data, may enable and facilitate several potential applications in wireless communications, extending beyond the localization task. Learned channel representations can serve as pseudo locations and aid various tasks, from beamforming and UE tracking to more user-centric location-based services (LBS). For instance, by evaluating a *distance* metric between the points in the feature space, one can determine if two transmitters are close to a reference location, or a *spot*. This can be advantageous for asset or user tracking, assisted indoor navigation, and targeted advertising, to highlight a few examples.

Formally, our goal for the rest of this dissertation is to learn a channel representation $\bar{\mathbf{o}}_r$ (i.e., an embedding), using neither labels nor a contrastive objective function. Hence, let us first construct a *backbone* neural network architecture that employs a dual encoder-projector framework. This framework aligns with the overall architectural and learning paradigm presented in [140, 142]. The encoders, represented by $f_{\boldsymbol{\Theta}}(\cdot)$ and $f_{\boldsymbol{\Psi}}(\cdot)$, and the projectors, denoted by $g'_{\boldsymbol{\Theta}}(\cdot)$ and $g'_{\boldsymbol{\Psi}}(\cdot)$, are parameterized by $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}$, respectively. Further, assuming two input representations of the same channel realization, $\mathbf{H}_r$ and $\mathbf{H}_r^{(+)}$ (e.g., two augmented views), we can write a common SSL formulation as

$$\{\boldsymbol{\Theta}^\star, \boldsymbol{\Psi}^\star\} = \arg\min_{\boldsymbol{\Theta},\boldsymbol{\Psi}} \mathcal{L}_{\text{SSL}} \left(g'_{\boldsymbol{\Theta}}(f_{\boldsymbol{\Theta}}(\mathbf{H}_r)), g'_{\boldsymbol{\Psi}}(f_{\boldsymbol{\Psi}}(\mathbf{H}_r^{(+)})) \right) , \tag{5.5}$$

where $\mathcal{L}_{\text{SSL}}(\cdot)$ is the SSL loss function, which we will detail later in this section. Having a set of optimal parameters $\boldsymbol{\Psi}^\star$, our goal then becomes to fine-tune the parameters of the encoder $f_{\boldsymbol{\Psi}^\star}(\cdot)$ (a.k.a. momentum encoder of the *backbone* network) altogether with the MLP head $g_{\boldsymbol{\Phi}}(\cdot)$ to learn a mapping function between the channel $\mathbf{H}_r$ and user location information, $\mathbf{u}_r$,

$$\begin{aligned} \{\boldsymbol{\Psi}^{\star\star}, \boldsymbol{\Phi}^\star\} &= \arg\min_{\boldsymbol{\Psi}^\star,\boldsymbol{\Phi}} \mathcal{L}_{\text{SUP}} \left(\mathbf{u}_r, g_{\boldsymbol{\Phi}}(f_{\boldsymbol{\Psi}^\star}(\mathbf{H}_r)) \right) \\ &= \arg\min_{\boldsymbol{\Psi}^\star,\boldsymbol{\Phi}} \mathbb{E} \left[\left\| \mathbf{u}_r - g_{\boldsymbol{\Phi}}(f_{\boldsymbol{\Psi}^\star}(\mathbf{H}_r)) \right\|^2 \right] . \end{aligned} \tag{5.6}$$

To illustrate the connection with the WiT, it can be easily observed that (5.6) is equivalent to fine-tuning WiT using the parameters from one of the encoders in (5.5). This, however, does not incorporate WiT's default MLP head. Instead, we introduce a new MLP head during the fine-tuning process. In contrast to the MLP head of WiT, which can incorporate non-linearities, here we employ a single linear layer MLP

head. This allows us to ensure that the useful and generalizable representations are learned by the encoder and not by the complexity of the task-specific head.

Alternatively, in case of linear evaluation, we aim to only train the linear MLP head on top of the *frozen* parameters of the encoder $f_{\mathbf{\Psi}^\star}(\cdot)$, and only update $\mathbf{\Phi}$, i.e.,

$$\mathbf{\Phi}^\star = \arg\min_{\mathbf{\Phi}} \mathcal{L}_{\text{SUP}}\left(\mathbf{u}_r, g_{\mathbf{\Phi}}(f_{\mathbf{\Psi}^\star}(\mathbf{H}_r))\right) . \tag{5.7}$$

Having established our objective and formalized the self-supervised wireless channel representation learning, we proceed to elaborate on the key components of SWiT.

5.4 SWiT: Self-Supervised Wireless Transformer

The main building blocks of SWiT are depicted in Fig. 5.4. In a nutshell, SWiT comprises a two-branch DNN, which we denote as online and target networks, respectively. In most parts, these two networks share the same architectural design. However, they have different sets of weights. Given a single input channel realization, we aim to derive a robust channel representation in the presence of perturbations, which we model as augmentations. To do so, we define $\mathcal{T}_i(\mathbf{H}_r)$ as the sequential application of transformations, where each transformation Q_a is applied with a certain probability, $\mathbb{P}(Q_a)$. Specifically,

$$\tilde{Q}_a = \begin{cases} Q_a & \text{with probability } \mathbb{P}(Q_a), \\ \text{Identity} & \text{otherwise}, \end{cases} \tag{5.8}$$

and $\mathcal{T}_i(\mathbf{H}_r) = \left(\tilde{Q}_A^{(i)} \circ \tilde{Q}_{A-1}^{(i)} \circ \ldots \circ \tilde{Q}_1^{(i)}\right)(\mathbf{H}_r)$. For each $i \in \{1,2,3\}$, $\mathcal{T}_i$ represents a unique combination of transformations occurring sequentially and according to their predefined probability. To illustrate, in the case of $\mathcal{T}_1$, the probability of Q_a occurring can be specifically set to zero. More specifically, the role of the stochastic augmentation module is to output two views of the channel comprising most of the subcarriers, i.e.,

$$\{\bar{\mathbf{h}}'_n \in \mathbb{R}^{3N_r}\}_{n=1}^{N'_{\text{cg}}} \triangleq \mathcal{T}_1\left(\mathbf{H}_r\right) \tag{5.9}$$

for the first view, and similarly for the second view,

$$\{\tilde{\mathbf{h}}'_n \in \mathbb{R}^{3N_r}\}_{n=1}^{N'_{\text{cg}}} \triangleq \mathcal{T}_2\left(\mathbf{H}_r\right) , \tag{5.10}$$

where N'_{cg} denotes the total number of subcarrier representations after resizing operation for the respective transformation. We detail the subcarrier selection, and individual transformations later in Sec.5.4.1. We will refer to these transformed channels as *global* views. Similarly, the augmentation module is also designed to produce N_s random channel views. This involves applying the transformation function

$\mathcal{T}_3$ to a subset of subcarriers. Specifically, we apply $\mathcal{T}_3$ repeatedly N_s times, each time randomly selecting a set of subcarriers and resizing by interpolation to a fixed N'_{cl} representations,

$$\{\hat{\mathbf{h}}'_n \in \mathbb{R}^{3N_r}\}_{n=1}^{N'_{\text{cl}}} \triangleq \mathcal{T}_3\left(\mathbf{H}_r\right) . \tag{5.11}$$

We refer to the respective augmented channels as *local* views. Given that we derive one view from the initial channel transformation, $\mathcal{T}_1$, another from the application of $\mathcal{T}_2$, and N_s more views from $\mathcal{T}_3$, the total number of views for the same input channel becomes $V = 2 + N_s$.

Both *global* and *local* channel views are processed consecutively through the online $f_{\boldsymbol{\Theta}}(\cdot)$ and target encoder $f_{\boldsymbol{\Psi}}(\cdot)$, accordingly producing the corresponding representations for a single view,

$$\{\bar{\mathbf{o}}_n \in \mathbb{R}^D\}_{n=1}^{C_{\text{cg}}} := f_{\boldsymbol{\Psi}}\left(\{\bar{\mathbf{h}}'_n\}_{n=1}^{N'_{\text{cg}}}\right) , \tag{5.12}$$

where $C_{\text{cg}} = N'_{\text{cg}} + 1$ (i.e., including the [LID]), and similarly for the other views.

After obtaining the representations from the encoders, we forward the subcarrier representations to two distinct projectors, namely, micro-fading and macro-fading level embeddings learning projectors. By introducing these two new projectors, we aim to capture better the large-scale channel characteristics along with small-scale, per-subcarrier features. Finally, we compute the loss on the final embeddings of views obtained from online and target networks. Prior to explaining the large-scale and small-scale feature learning components of SWiT, depicted in Fig. 5.4, we provide further details on the individual stochastic channel transformations.

5.4.1 Stochastic Channel Augmentations

We rely on pretext tasks to learn robust channel representations. We formulate such tasks with the idea of ensuring that, during the learning phase, the resulting representations should remain invariant to their augmented views or channel transformations. Hence, the design of augmentations, or view-selection function $\mathcal{T}_i(\cdot)$, is an essential aspect influencing learned invariances that yield useful channel representations. As we mentioned in the previous section, the augmentation module's role is to transform an input channel realization into a pre-defined number of correlated views. Specifically, we sequentially apply the following channel transformations. We first apply *random subcarrier selection (RSS)* followed by resizing of the grouped subcarriers to a fixed size, and optionally apply *random subcarrier flipping (RSF)*. Furthermore, we apply a *random gain offset (RGO)*, followed by a *random fading component (RFC)*, *random sign change (RSC)*, *normalization*, and optionally we add *Gaussian noise*.

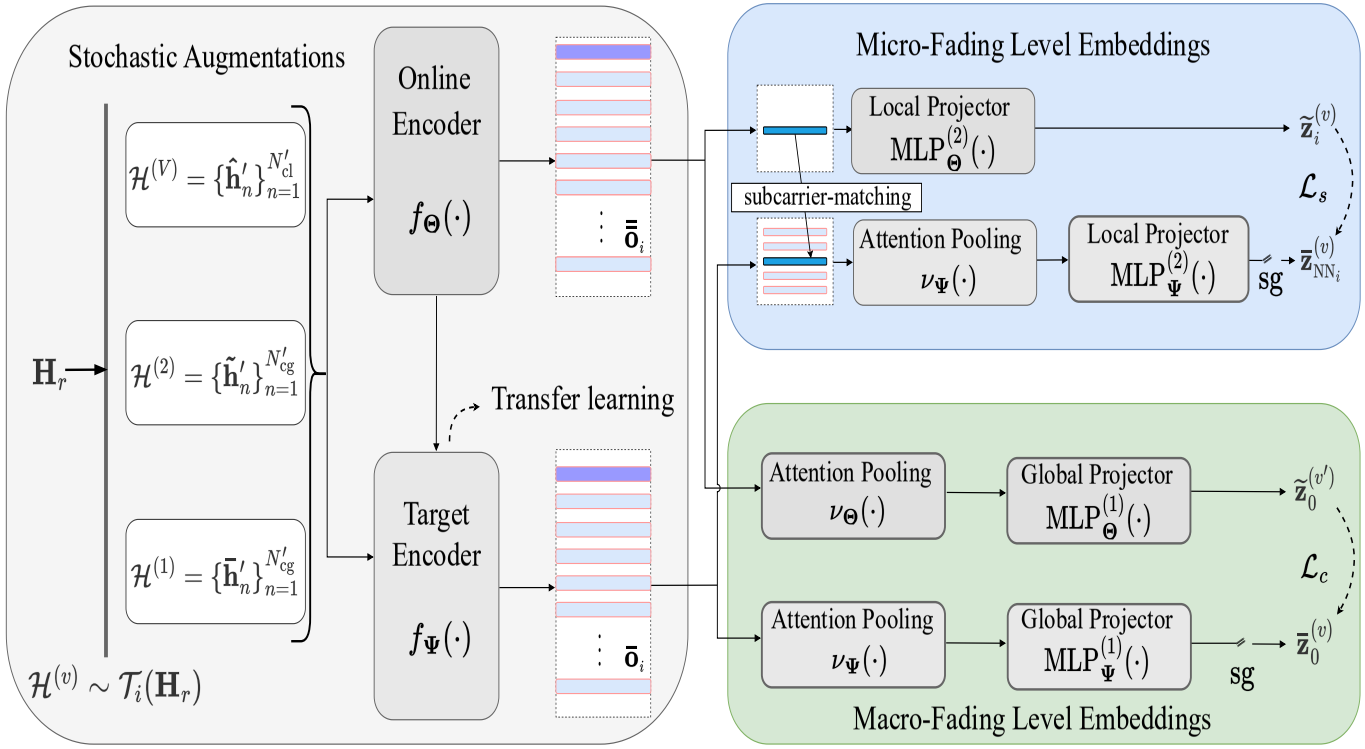


Figure 5.4: Main components of SWiT. Target encoder serves for transfer learning.

Random Subcarrier Selection (RSS)

As already stated, the networks receive inputs in the form of two *global* views and N_s *local* views. Each of these views comprises a selected number of subcarriers. We select $N'_{c_i} = \lfloor \gamma_{c_i} N'_c \rfloor$ adjacent subcarriers and, for practical reasons, linearly interpolate to resize the whole channel to a fixed size $N_r \times N'_{\text{cg}}$, i.e., N'_{cg} subcarrier representations for the input to the encoders for *global* views. Similarly, for the *local* views, we set a fixed size of $N_r \times N'_{\text{cl}}$ before feeding them to the online and target encoders. Unless mentioned otherwise, we select $\gamma_{c_1} = 0.9$, and $\gamma_{c_2} = 0.8$, for the *global* views and $\gamma_{c_3} = 0.1$ for the *local* views. We define $\gamma_{c_1}, \gamma_{c_2}$, and γ_{c_3} when constructing $\mathcal{T}_1, \mathcal{T}_2$ and $\mathcal{T}_3$. By doing so, i.e., feeding the model with channel views at various scales, we force the network to learn the relationship between the channel features between the individual subcarriers and among the representations of the multiple subcarriers.

Random Subcarrier Flipping (RSF)

We optionally apply *mirroring*, where each subcarrier representation is flipped as

$$Q_a(\mathbf{H}_r) = \mathbf{H}_r \mathbf{P}_{\text{flip}}\,, \tag{5.13}$$

where $\mathbf{P}_{\text{flip}}$ has the size of $N'_{\text{cg}} \times N'_{\text{cg}}$, with elements

$$p_{\text{flip}_{i,j}} = \begin{cases} 1 & \text{if } j = N'_{\text{cg}} - (i-1) \\ 0 & \text{otherwise} \end{cases}\,. \tag{5.14}$$

We commonly apply *RSF* on both *global* views. For all other N_s views, the probability of *RSF* being applied to *local* views is set to zero.

Random Gain Offset (RGO)

Selecting a subset of subcarriers alone yields representations that may share the same channel gain distribution between antenna elements at any two subcarriers, enabling the model to exploit this property and quickly minimize the loss. However, by doing so, the network may fail to capture generalizable channel characteristics. To circumvent such a phenomenon, we scale instantaneous channel coefficients by a constant offset, $\xi_{\text{o}} = 1 \pm \xi_{\text{rgo}}$,

$$Q_a(h_{n,i}) = \xi_{\text{o}} h_{n,i} \qquad \forall i = 1, \ldots, 3N_r\,. \tag{5.15}$$

Through experimentation, we found that $\xi_{\text{rgo}} \sim \mathcal{U}(0, 0.1)$ is sufficient to avoid identical channel views during the self-supervised training.

Random Fading Component (RFC)

To achieve robustness due to additional multipath components, we shift the instantaneous peak power of the channel. We do so by applying a Rayleigh distribution only to the absolute part of the transformed view,

$$Q_a(h_{n,i}) = \frac{h_{n,i}}{\sigma^2_{\text{rfc}}} \exp\left(-h^2_{n,i} / \left(2\sigma^2_{\text{rfc}}\right)\right) \quad \forall i = 2N_r + 1, \ldots, 3N_r \,. \tag{5.16}$$

Here, we find the scale factor $\sigma_{\text{rfc}} \sim \mathcal{U}(0.5, 0.6)$ through experimentation. Furthermore, to help the model improve its ability to handle uncertainty, we apply *RFC* only to the second view, denoted as $\mathcal{H}^{(2)}$ in Fig. 5.4.

Random Sign Change (RSC)

We randomly negate all real-valued channel coefficients for the second view (i.e., $\mathcal{H}^{(2)}$),

$$Q_a(h_{n,i}) = (-1)h_{n,i} \qquad \forall i = 1, \ldots, 3N_r. \tag{5.17}$$

This technique can be understood as a form of introducing an adversarial example, thereby increasing the robustness of the model.

Normalization

We finally post normalize all the channel transformations by dividing real, imaginary and magnitude parts with their corresponding maximum absolute values, $\Delta_{\text{Re}} = \max(\max(\{|\mathbf{H}^{(\text{Re})}_r|\}^R_{r=1}))$. Similarly, we normalize the imaginary, Δ_{Im}, as well as the absolute part, Δ_{Abs}.

Gaussian Noise

Also, we occasionally inject additional randomness by adding Gaussian noise to the normalized channel,

$$Q_a(h_{n,i}) = h_{n,i} + \omega_{n,i} \,, \tag{5.18}$$

where $\omega_{n,i}$ is zero-mean Gaussian noise with variance $\sigma^2_Q = 1.0 \times 10^{-7}$.

In Table 5.2 we present the default augmentations selected for $\mathcal{T}_1(\cdot)$, $\mathcal{T}_2(\cdot)$, and $\mathcal{T}_3(\cdot)$, along with their corresponding assigned probabilities, $\mathbb{P}(Q_a)$.

Table 5.2: Transformation function $\mathcal{T}_i(\cdot)$ and $\mathbb{P}(Q_a)$.

Q_a , $\mathbb{P}(Q_a)$	$\mathcal{T}_1(\cdot)$	$\mathcal{T}_2(\cdot)$	$\mathcal{T}_3(\cdot)$
Subcarrier selection (RSS)	1.0	1.0	1.0
Subcarrier flipping (RSF)	0.4	0.4	0.0
Gain offset (RGO)	0.2	0.8	0.0
Fading component (RFC)	0.0	0.1	0.0
Sign change (RSC)	0.0	0.2	0.0
Normalization	1.0	1.0	1.0
Gaussian noise	0.2	0.2	0.0

It is worth clarifying that even though $\mathcal{T}_3$ may use stochastic transformations with a probability set to one, applying it repeatedly to the estimated channel matrix $\mathbf{H}_r$ will not necessarily output identical transformed channels. Given, for instance, $\gamma_{c_3} = 0.1$, *RSS* randomly selects just a few subcarriers. This random selection ensures varied subcarrier subsets when $\mathcal{T}_3$ is applied, resulting in potentially different *local* views.

5.4.2 Macro-fading Level Representations

We train SWiT in a self-supervised manner, wherein the model is tasked with predicting channel views that are transformed to reveal different characteristics in the embedding spaces of both the online and target networks. This approach relies on the idea that by learning to differentiate and represent such characteristics, the model achieves a more general *understanding* of beneficial channel features for wireless localization and other downstream tasks.

Specifically, the online network processes both *local* as well as *global* channel views (i.e., $V = 2 + N_s$ views) to output the respective embeddings

$$\widetilde{\mathbf{z}}_0^{(v')} = \frac{\exp(\dot{\mathbf{z}}_0^{(v')}\chi_{\boldsymbol{\Theta}}^{-1})}{\sum_{j=1}^{n}\exp(\dot{z}_{0,j}^{(v')}\chi_{\boldsymbol{\Theta}}^{-1})}\,, \tag{5.19}$$

where $v' \in \{1, \ldots, 2 + N_s\}$ has been added to make the index of the channel's views explicit. In (5.19), $\chi_{\boldsymbol{\Theta}}$ is a *temperature* scaler that controls the peak of the output distribution for the online network. The embeddings $\dot{\mathbf{z}}_0^{(v')}$ are mapped from the global projector as

$$\dot{\mathbf{z}}_0^{(v')} = \mathrm{MLP}_{\boldsymbol{\Theta}}^{(1)} \circ \nu_{\boldsymbol{\Theta}}(\{\bar{\bar{\mathbf{o}}}_n^{(v')}\}_{\forall n})\,, \tag{5.20}$$

where n spans the range of subcarrier representations, with the total number of representations being either N'_{cg} for global views or N'_{cl} for local views. The intermediate function, $\nu(\cdot)$, is denoted as attention pooling operation, implemented as

a transformer block, and $\bar{\bar{\mathbf{o}}}_n^{(v')} \in \mathbb{R}^D$ is the output from the online encoder. On the other hand, for the target neural network branch, we evaluate

$$\bar{\mathbf{z}}_0^{(v)} = \frac{\exp((\mathring{\mathbf{z}}_0^{(v)} - \boldsymbol{\epsilon}_0)\chi_{\boldsymbol{\Psi}}^{-1})}{\sum_{j=1}^{n} \exp((\mathring{z}_{0,j}^{(v)} - \epsilon_{0,j})\chi_{\boldsymbol{\Psi}}^{-1})} \,. \tag{5.21}$$

Likewise, $\chi_{\boldsymbol{\Psi}}$ is the *temperature* scaler for the target model that controls the peak of the output distribution. Parameter $\boldsymbol{\epsilon}$ is used for centering of the ouput distribution, and we later detail it in Sec. 5.4.3. The embeddings $\mathring{\mathbf{z}}_0^{(v)}$ are mapped from the global projector of the target network as

$$\mathring{\mathbf{z}}_0^{(v)} = \mathtt{sg}(\mathrm{MLP}_{\boldsymbol{\Psi}}^{(1)} \circ \nu_{\boldsymbol{\Psi}}(\{\bar{\mathbf{o}}_n^{(v)}\}_{\forall n})) \,, \tag{5.22}$$

where `sg` is the `stop-gradient` operation.

To match the distribution between the embedding outputs of the target and online networks across different views, we compute the cross-entropy to compare each *global* view from the target encoder with any other alternate view from the online encoder,

$$\mathcal{L}_c := -\frac{1}{2(V-1)} \sum_{v=1}^{2} \sum_{v' \neq v}^{V} \bar{\mathbf{z}}_0^{(v)} \log\left(\tilde{\mathbf{z}}_0^{(v')}\right) \,. \tag{5.23}$$

5.4.3 Micro-fading Level Representations

Alongside capturing the channel's macroscopic fading characteristics, we introduce a pretext task that can be essential to leverage the small-scale fading of the channel. To achieve this, we aim to learn subcarrier-level representations, as depicted in the upper-right part of Fig. 5.4. Specifically, we assume neighboring subcarriers as positive examples. By doing so, i.e., performing a limited neighborhood search, we maintain lower computational complexity, as opposed to considering all subcarrier representations. Therefore, for each embedding $\bar{\mathbf{o}}_i^{(v)}$, we first evaluate its correlation with the representations in its neighborhood $\{\bar{\mathbf{o}}_j^{(v)}\}_{j \in \mathcal{N}_i}$, where $\mathcal{N}_i$ is the set of indices of adjoint subcarrier-representations, $|\mathcal{N}_i| = K_n$, and $K_n \ll N_c'$. We sort the indices based on the similarity values in the descending order,

$$\mathcal{N}_i' = \underset{\substack{K_n \\ j \in \mathcal{N}_i}}{\mathrm{sort}} \max \rho\left(\bar{\mathbf{o}}_i^{(v)}, \bar{\mathbf{o}}_j^{(v)}\right) , \tag{5.24}$$

where

$$\rho(\bar{\mathbf{o}}_i^{(v)}, \bar{\mathbf{o}}_j^{(v)}) = \frac{\langle \bar{\mathbf{o}}_i^{(v)}, \bar{\mathbf{o}}_j^{(v)} \rangle}{\left\| \bar{\mathbf{o}}_i^{(v)} \right\| \left\| \bar{\mathbf{o}}_j^{(v)} \right\|} \tag{5.25}$$

is the measure of similarity, i.e., the correlation between $\bar{\mathbf{o}}_i^{(v)}$ and embeddings in its neighbourhood $\mathcal{N}_i$. Finally, we select $K_k = |\mathcal{P}_i|$, where $\mathcal{P}_i \subset \mathcal{N}_i'$, represents the `top-k` matched indices.

Instead of fusing (e.g., averaging) `top-k` matched subcarrier-representations, we pass $\{\bar{\mathbf{o}}_j^{(v)}\}_{j\in\mathcal{P}_i}$ to another transformer block, and output a compressed representation $\mathbf{z}_{\mathrm{NN}_i}^{(v)}$,

$$\mathbf{z}_{\mathrm{NN}_i}^{(v)} := \nu_{\mathbf{\Psi}}(\{\bar{\mathbf{o}}_j^{(v)}\}_{j\in\mathcal{P}_i})\,. \tag{5.26}$$

We feed the small-scale embeddings through online and target feedforward MLP networks $\mathrm{MLP}_{\mathbf{\Theta}}^{(2)}(\cdot)$ and $\mathrm{MLP}_{\mathbf{\Psi}}^{(2)}(\cdot)$, respectively. We denote the corresponding embeddings as $\dot{\mathbf{z}}_i^{(v)} := \mathrm{MLP}_{\mathbf{\Theta}}^{(2)}(\bar{\bar{\mathbf{o}}}_i^{(v)})$ and $\mathring{\mathbf{z}}_{\mathrm{NN}_i}^{(v)} := \mathtt{sg}(\mathrm{MLP}_{\mathbf{\Psi}}^{(2)}(\mathbf{z}_{\mathrm{NN}_i}^{(v)}))$, respectively. Finally, same as we evaluated (5.19) and (5.21) for the macroscopic learning part, we also evaluate $\tilde{\mathbf{z}}_i^{(v)}$ for the online and $\bar{\mathbf{z}}_{\mathrm{NN}_i}^{(v)}$ for the target neural networks of the microscopic learning part. The loss function for learning the micro-fading channel characteristics is calculated over all the subcarrier reprentations N', where N' can be either N_{cg}' or N_{cl}' for each view, and is written as

$$\mathcal{L}_s := -\frac{1}{VN'}\sum_{v=1}^{V}\sum_{i=1}^{N'}\bar{\mathbf{z}}_{\mathrm{NN}_i}^{(v)}\log\left(\tilde{\mathbf{z}}_i^{(v)}\right)\,. \tag{5.27}$$

Finally, the total loss function is computed as $\mathcal{L}_{\mathrm{SSL}} := \mathcal{L}_c + \beta\mathcal{L}_s$, where $\beta \in [0,1]$ controls the relevance of learning macroscopic fading versus microscopic fading.

Optimization

In general, given the same input channel realization, multi-branch network architectures (e.g., based on Siamese as in Chapter 3) may suffer mode collapse, resulting in both the target and online networks outputting the same constant, hence, demanding careful negative pair mining. In our case, we avoid such minima while forsaking the need for contrastive loss. We rely on expontial moving averaging (EMA) to build and update the target network parameters as in [140]. Thus, the gradients do not backpropagate through the target network with $\mathbf{\Psi}$, as shown by the `stop-gradient` (`sg`) operator in Fig. 5.4. Instead, given a target decay rate $\kappa \in [0,1]$, the set of parameters $\mathbf{\Psi}$ is only updated via the online network after each training step as $\mathbf{\Psi} \leftarrow \kappa\mathbf{\Psi} + (1-\kappa)\mathbf{\Theta}$. Finally, to further enhance the learning, minimize the dependency on batch size, as well as batch normalization, we perform *sharpening* and *centering* of the output features distribution, and freeze the target network over the first epoch, as it is suggested in [142]. More precisely, *centering* is viewed as adding a bias term $\boldsymbol{\epsilon}$ to the target network, $g_{\mathbf{\Psi}}' \circ f_{\mathbf{\Psi}}(\mathbf{H}_i) \leftarrow g_{\mathbf{\Psi}}' \circ f_{\mathbf{\Psi}}(\mathbf{H}_i) + \boldsymbol{\epsilon}$, and is

updated with EMA as

$$\boldsymbol{\epsilon} \leftarrow \Lambda\boldsymbol{\epsilon} + (1-\Lambda)\frac{1}{B}\sum_{i=1}^{B} g'_{\boldsymbol{\Theta}} \circ f_{\boldsymbol{\Theta}}(\mathbf{H}_i)\,. \tag{5.28}$$

At the end of training, we only utilize the target encoder, $f_{\boldsymbol{\Psi}^\star}(\cdot)$, with learned parameters $\boldsymbol{\Psi}^\star$. We outline the overall self-supervised training procedure in Algorithm 1. The self-supervised training details and the evaluation procedure are detailed in the next section.

Algorithm 1 Self-supervised Training

Given: A dataset of unlabeled channel estimates $\{\mathbf{H}_r\}_{r=1}^{R}$, transformation function $\mathcal{T}_i = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3\}$, the `AdamW` optimizer, U optimization steps, batch size B, learning rate values $\{\varpi\}_{u=1}^{U}$, and schedule rates $\{\kappa\}_{u=1}^{U}$

Build $f_{\boldsymbol{\Theta}}, f_{\boldsymbol{\Psi}}, \nu_{\boldsymbol{\Theta}}, \nu_{\boldsymbol{\Psi}}, \text{MLP}_{\boldsymbol{\Theta}}^{(1)}, \text{MLP}_{\boldsymbol{\Theta}}^{(2)}, \text{MLP}_{\boldsymbol{\Psi}}^{(1)}, \text{MLP}_{\boldsymbol{\Psi}}^{(2)}$, and initialize $\boldsymbol{\Theta}, \boldsymbol{\Psi}$

for $u \leftarrow 1$ to U **do**
  Sample $\mathcal{B} \leftarrow \{\mathbf{H}_i\}_{i=1}^{B}$
  for $\mathbf{H}_i \in \mathcal{B}$ **do**
    V views according to (5.9), (5.10), and (5.11)
    $\dot{\mathbf{z}}_0^{(v)} \leftarrow \text{MLP}_{\boldsymbol{\Theta}}(\nu_{\boldsymbol{\Theta}}(f_{\boldsymbol{\Theta}}(\mathcal{T}_i(\mathbf{H}_i))))$ $\{V$ views$\}$
    $\mathring{\mathbf{z}}_0^{(v)} \leftarrow \text{MLP}_{\boldsymbol{\Psi}}(\nu_{\boldsymbol{\Psi}}(f_{\boldsymbol{\Psi}}(\mathcal{T}_i(\mathbf{H}_i))))$ $\{V$ views$\}$
    $\dot{\mathbf{z}}_i^{(v)} \leftarrow \text{MLP}_{\boldsymbol{\Theta}}(f_{\boldsymbol{\Theta}}(\mathcal{T}_i(\mathbf{H}_i)))$ $\{V$ views$\}$
    $\mathring{\mathbf{z}}_{\text{NN}_i}^{(v)} \leftarrow \text{MLP}_{\boldsymbol{\Psi}}(\nu_{\boldsymbol{\Psi}}(f_{\boldsymbol{\Psi}}(\mathcal{T}_i(\mathbf{H}_i))))$ $\{V$ views$\}$
    Compute $\widetilde{\mathbf{z}}_0^{(v)}$ and $\widetilde{\mathbf{z}}_i^{(v)}$ according to (5.19)
    Compute $\bar{\mathbf{z}}_0^{(v)}$ and $\bar{\mathbf{z}}_{\text{NN}_i}^{(v)}$ according to (5.21)
    Compute (5.23) for macroscopic and (5.27) for microscopic fading level features
    Compute $\mathcal{L}_{\text{SSL}}^{(i)} := \mathcal{L}_c + \beta\mathcal{L}_s$
  end for
  $\nabla_{\boldsymbol{\Theta}} \leftarrow \frac{1}{B}\partial\mathcal{L}_{\text{SSL}}^{(i)}$
  Update online parameters $\boldsymbol{\Theta} \leftarrow \texttt{AdamW}(\boldsymbol{\Theta}, \nabla_{\boldsymbol{\Theta}}, \varpi)$
  Update target parameters $\boldsymbol{\Psi} \leftarrow \kappa\boldsymbol{\Psi} + (1-\kappa)\boldsymbol{\Theta}$
  Update centering value according to (5.28)
end for

Return: Encoder $f_{\boldsymbol{\Psi}^\star}(\cdot)$ and learned parameters $\boldsymbol{\Psi}^\star$

5.5 Linear Evaluation and Fine-Tuning

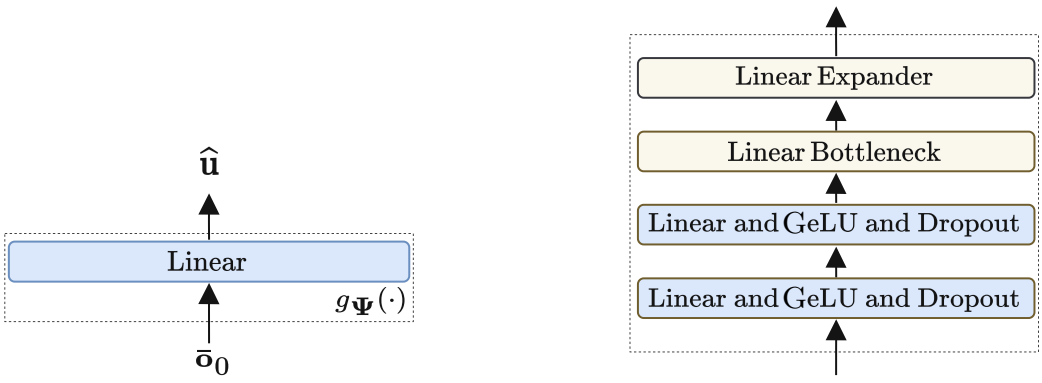
Before detailing the evaluation approach, we first describe the datasets and SSL training details. For the rest of this chapter, we utilize three different types of datasets to evaluate the performance of self-supervised channel features. These

datasets represent a range of environments and configuration settings, including indoors and outdoors, co-located massive MIMO systems, as well as DAS:

1) **KUL Dataset** - We utilize the *ultra dense indoor MaMIMO* data pool from [34], which includes multiple datasets characterized by different antenna configurations and propagation characteristics, all obtained in a laboratory environment. Specifically, in our study, we use three datasets: the first with NLOS propagation and a BS having a uniform rectangular array (KUL-NLOS-URA-Lab), the second with LOS propagation and a uniform linear array (KUL-LOS-ULA-Lab), and the third featuring a LOS environment but with a distributed antenna system setup (KUL-LOS-DIS-Lab). Across all the scenarios, the default configuration parameters are $N_r = 64$, $N'_c = 100$, $f_c = 2.61$GHz, and the sample size is $R = 250\,000$.

2) **S and HB Dataset** - Recalling their key characteristics, the S-200 and HB-200 datasets, detailed earlier in Sec. 5.2, represent ray-tracing based channel realizations in two railway scenarios, each with $T = 200$ snapshots. The S-scenario has $M = 1$ node and $R = 69\,212$ samples, whereas the HB-200 scenario comprises $M = 8$ nodes and $R = 81\,200$ samples. For both datasets we keep $N_r = 64$, $f_c = 3.5$GHz, number of paths $G = 4$, and $N'_c = 32$.

3) **WILD Dataset** - We utilize the WILD-v2 dataset [35]. This dataset consists of CSI samples from two *similar-looking* indoor environments, denoted as Env-1 and Env-2. Each environment is 40m $\times$ 20m in size, and there are $R = 28\,000$ and $R = 5\,000$ samples, respectively. In both scenarios, there are $M = 6$, and $f_c = 5.21$GHz, $N'_c = 234$, $N_r = 24$. For testing, we use $4\,000$ samples from Env-1 and $1\,000$ samples from Env-2, forming a combined test set of $R_{\text{test}} = 5\,000$. However, we only use Env-1 for pre-training and fine-tuning.



(a) MLP head.

(b) MLPs for local and global *projectors*.

Figure 5.5: (a) Task-specific MLP-head and (b) projector DNNs useful for SWiT.

Self-supervised Training Details

For both online and target network components, i.e., $f_{\boldsymbol{\Phi}}(\cdot)$, $f_{\boldsymbol{\Psi}}(\cdot)$, $\nu_{\boldsymbol{\Phi}}(\cdot)$, and $\nu_{\boldsymbol{\Psi}}(\cdot)$, we adopt the architectural details from the WiT model as presented at the beginning of this chapter, except for the MLP head. The base configuration includes one transformer block $H_{\text{blck}} = 1$ and a single attention head $H_{\text{attn}} = 1$. Since [78] has become a quasi-norm among modern transformer-based models across the fields, we follow their approach, and our implementation of the encoder in SWiT uses a dimensionality of $D = 384$, different from our past preference of $D = 650$ in WiT. The LayerNorm parameters are $\zeta = 1$ and $\iota = 0.0001$. We set $N_s = 8$, $N'_{\text{cg}} = 36$, $N'_{\text{cl}} = 16$, $K_n = 6$, and $K_k = 3$. Training SWiT typically involves several hundred epochs, with a common setting of about 500 epochs using a batch size of $B = 512$. However, certain scenarios and tasks show satisfactory performance with significantly fewer epochs, around $45 - 50$, and a reduced batch size of $B = 256$ in a single GPU works well. Given the significant time complexity involved, we train SWiT using a downsampled version of the datasets, roughly $10 - 25\%$ of a KUL dataset.

As a solver, we use AdamW [143] with a cosine schedule for the learning rate, ϖ, without restarts and a linear warm-up for 10 epochs, and fix $\beta = 0.1$. The base learning rate is $\varpi_{\text{base}} = 1.5 \times 10^{-4}$ and updates as $\varpi \triangleq \varpi_{\text{base}} B/256$. When employing a single GPU to train SWiT, there might be a need to recalibrate weight decay and learning rate parameters, depending on the dataset size and number of iterations (i.e., the batch size). The target network updates use $\kappa_{\text{base}} = 0.994$ updated with $\kappa \triangleq 1 - (1 - \kappa_{\text{base}})\,(\cos(\pi u/U) + 1)/2$. We use weight decay scaled from 0.04 to 0.5, set $\chi_{\boldsymbol{\Psi}} = 0.04$, $\chi_{\boldsymbol{\Theta}} = 0.1$, and $\Lambda = 0.9$. During the first epoch, the target network is frozen. The MLPs for local and global projectors are constructed with four layers, where the initial two are linear layers followed by GeLU non-linearity and comprise 1 024 units each. We observe that removing one of these first two layers does not diminish performance; in fact, it may even accelerate convergence, particularly evident in training on the KUL dataset. The third layer is a linear bottleneck with 256 neurons, and the last layer (i.e., the linear expander) has 25 000 neurons for $\text{MLP}^{(1)}(\cdot)$ and 1 024 neurons for $\text{MLP}^{(2)}(\cdot)$. For the reference, task-specific MLP head and the MLPs for the projectors are depicted in Fig. 5.5.

After the models are trained, we evaluate the performance of the learned representations using the standard evaluation protocol in ML for SSL. Specifically, we conduct linear evaluation and fine-tuning of the trained models to assess the usefulness of learned channel features for UE location estimation. Additionally, we extend our investigation to examine the models' transferability across various propagation settings and their applicability to other predictive tasks, e.g., pathloss estimation.

Linear Evaluation Approach

For linear evaluations, we train a linear regressor, $g_{\mathbf{\Phi}} : \mathbb{R}^D \mapsto \mathbb{R}^{D_{\text{out}}}$, (or a classifier) on top of the *frozen* features from $f_{\mathbf{\Psi}^\star}(\cdot)$, as denoted in (5.7). In the case of the regressor and the localization task, $D_{\text{out}} = 2$, i.e., location coordinates, since $u_{i,3}$ is a constant value in the datasets we utilize. For a just evaluation, we apply no augmentations during the fine-tuning, and if not mentioned otherwise, we report the accuracy on $N_c' = 32$ subcarriers. We commonly apply normalization, although learned embeddings are less sensitive to first-order statistics. For linear regressor, we train a one-layer MLP for 500 epochs with $B = 128$ and AdamW with a fixed $\varpi = 3 \times 10^{-4}$. Alternatively, one can use the SGD with $\varpi = 0.03$. We also assess the quality of embeddings with a non-parametric method, i.e., the weighted nearest neighbor classifier (k-NN). Similarly, we *freeze* the pre-trained model $f_{\mathbf{\Psi}^\star}(\cdot)$, and then compute and store the channel representations. Then, we apply k-NN to match the features of the input channel, and report `top-1`, and when relevant, `top-5` accuracy. The `top-1` accuracy is calculated as the proportion of correct predictions over total predictions and `top-5` accuracy allows the model to output the five most likely predictions.

Fine-Tuning Evaluation Approach

We evaluate the performance of the embeddings by fine-tuning the models with labeled samples. More specifically, we add a randomly initialized MLP head with a hidden linear layer on top of the encoder, $g_{\mathbf{\Phi}} : \mathbb{R}^D \mapsto \mathbb{R}^{D_{\text{out}}}$, initialize the backbone (i.e., the target encoder) with our pre-trained weights, and train $g_{\mathbf{\Phi}} \circ f_{\mathbf{\Psi}^\star}(\mathbf{H})$. We use $B = 512$, $\varpi = 3 \times 10^{-4}$, and AdamW with default parameter values. Like in linear evaluation setup, the accuracy is reported on 32 subcarriers after normalization. Since our method is batch normalization-free, it is worth noting that during the testing, we use $B = 1$. While we use no labels for self-supervised training, during the fine-tuning and supervised evaluation we apply stratified sampling without replacement.

5.5.1 Localization and Spot Estimation

In Fig. 5.6, we present the performance analysis with varying numbers of training samples, based on the datasets described in the earlier section. This evaluation encompasses three distinct setups, i.e., when employing a linear regressor, fine-tuning the SWiT target encoder, and training a fully-supervised model with randomly initialized weights, i.e., WiT. For the data regimes, we use $R = \{1\,000, 5\,000, 10\,000\}$ for training and $R_{\text{test}} = 5\,000$ without replacement for testing. For ease of reference, these configuration setups are labeled in the reporting figures and tables as 'Lin.', 'Fin.' and 'Sup.', respectively. As can be seen from the figure, the pre-trained models outperform the fully-supervised case in *all* three data regimes. We observe that when

a very small amount of training samples is available, in certain conditions, we can outperform the fully-supervised training even with a linear regressor. This behavior is especially observed in Figs. 5.6a and 5.6f, i.e., for KUL-NLOS and WILD-v2 datasets. Furthermore, to illustrate the impact of normalization, for instance, in Fig. 5.6a, we show that the linear regressor trained on self-supervised embeddings is more robust than fully-supervised encoder without normalization when $R = 1\,000$.

Spot Estimation

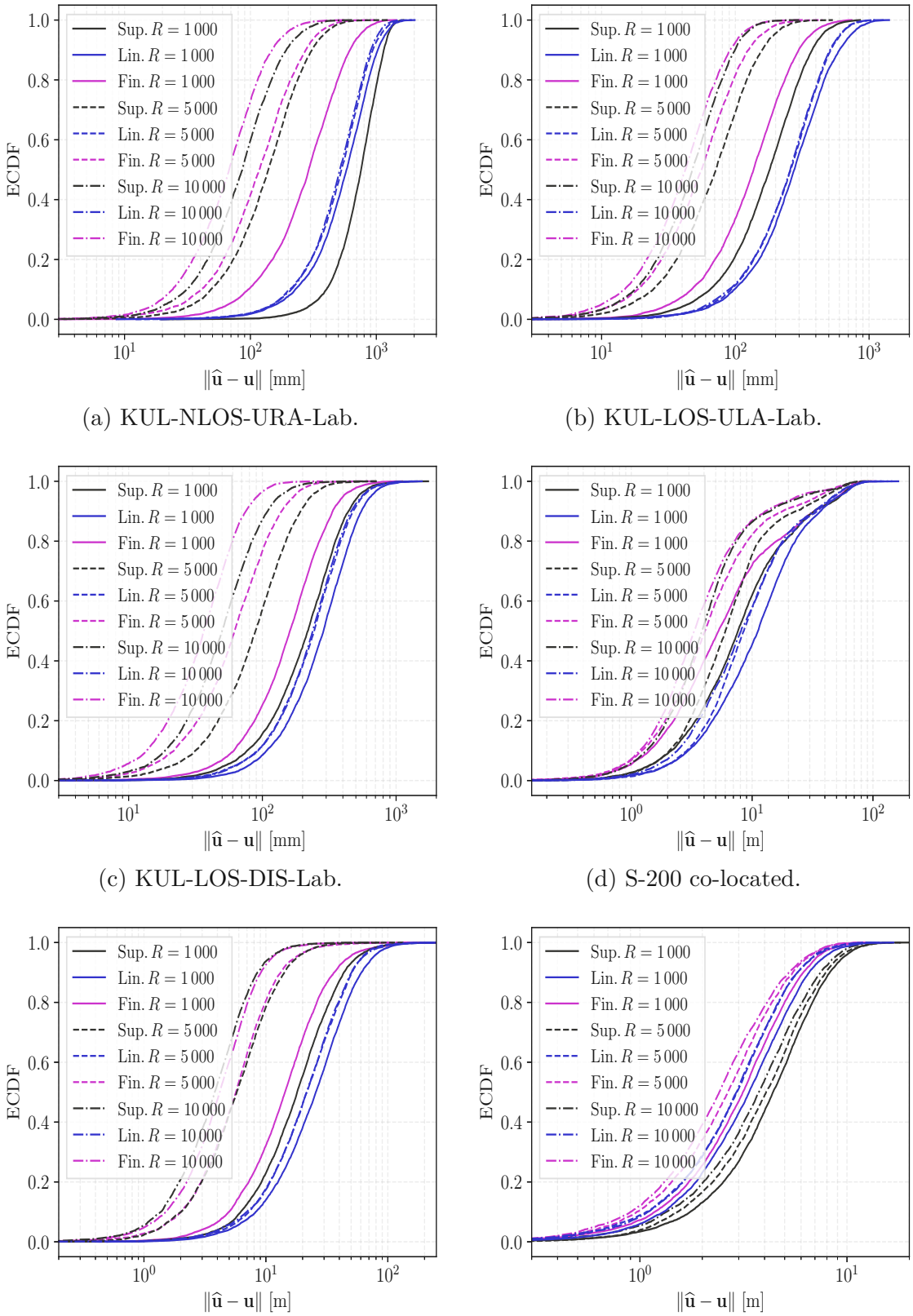
We investigate spot-localization, i.e., the ability of the model to *cluster* correlated channel representations while preserving the spatial neighborhood. To do so, we consider KUL, S-200, and HB-200 datasets. Since datasets in KUL are obtained from a scenario that is divided into four sub-regions, we consider $\widehat{C}_{\text{spot}} = 4$ spots. On the other hand, for S-200 and HB-200 datasets, we have $\widehat{C}_{\text{spot}} = 360$ and $\widehat{C}_{\text{spot}} = 406$, respectively. *Spot* localization can be especially beneficial in NLOS for the BS to perform context-aware beamforming and user tracking. For instance, the observed channel embedding may be compared with a reference *spot* embedding to detect the user and with a prior channel realization to track a moving user, all without needing actual location coordinates. In Table 5.3, we can read that embeddings from SWiT encoder can achieve near perfect accuracy for KUL datasets. This evaluation procedure clearly indicates that the pre-trained model yields remarkably better representations compared to embeddings of a randomly initialized WiT.

Table 5.3: Random weights versus SWiT.

	KUL-NLOS	KUL-LOS	KUL-LOS-DIS	S-200		HB-200	
Method	↑ Top-1	↑ Top-1	↑ Top-1	↑ Top-1	↑ Top-5	↑ Top-1	↑ Top-5
Random	23.7	3.59	4.13	0.296	1.545	0.27	1.25
SWiT	99.99	99.99	99.99	18.70	52.38	37.38	82.85

In Fig. 5.7, in the interest of visualization, we depict the computed two-dimensional t-SNE [144] embeddings of the individual datasets of KUL. We can easily observe that for the LOS propagation conditions, randomly initialized WiT and pre-trained models can yield qualitatively equal. However, when in NLOS, SWiT significantly improves the quality of representations. Each color in the figures represents a class (i.e., a *spot*).

In Appendix D.2, we present two-dimensional t-SNE embeddings of a merged KUL dataset, which combines CSI samples from LOS and NLOS datasets, highlighting the model's ability to differentiate between spots and propagation environments.



(a) KUL-NLOS-URA-Lab.

(b) KUL-LOS-ULA-Lab.

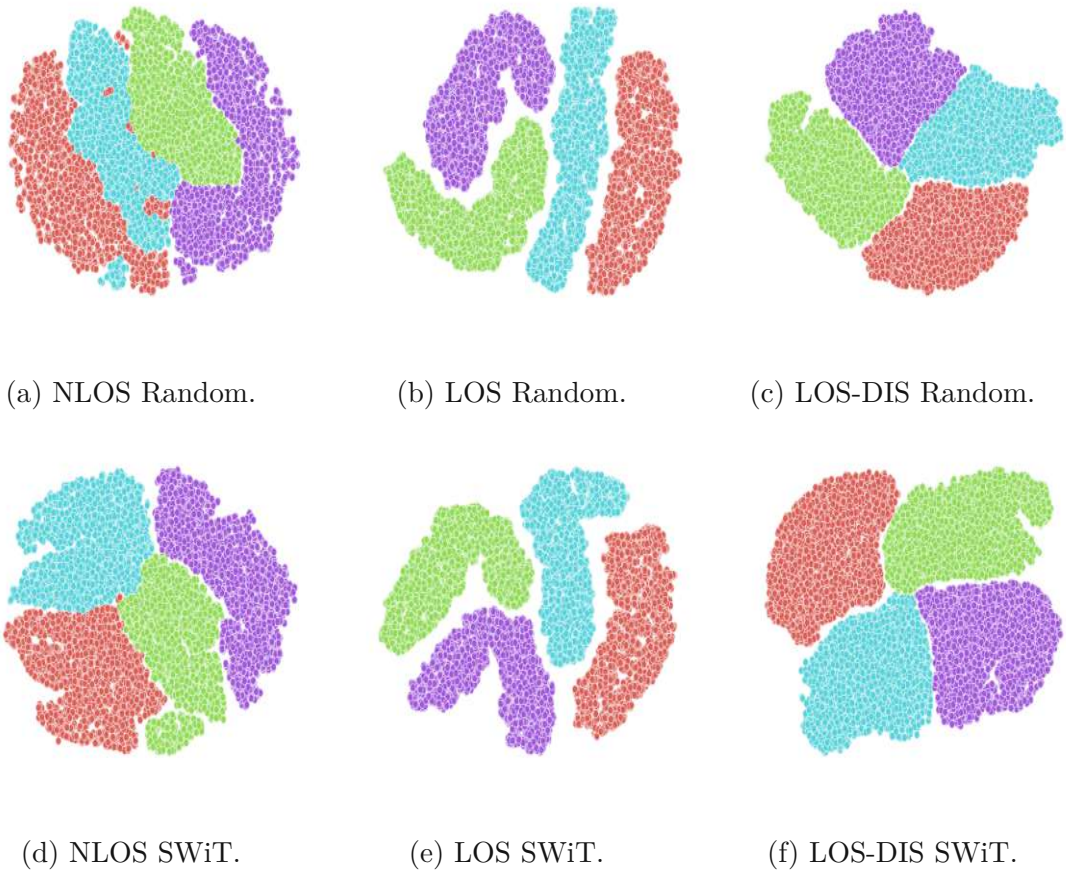
(c) KUL-LOS-DIS-Lab.

(d) S-200 co-located.

(e) HB-200 DIS.

(f) WILD DIS.

Figure 5.6: Direct localization with self-supervised channel features.



(a) NLOS Random. (b) LOS Random. (c) LOS-DIS Random.

(d) NLOS SWiT. (e) LOS SWiT. (f) LOS-DIS SWiT.

Figure 5.7: Two-dimensional t-SNE embeddings for the individual datasets.

Other Works

There has been a great amount of DNN-based localization techniques proposed over the years. Regardless, no such approach operates on self-supervised channel features. Nonetheless, we compare the localization performance of our method to the reported results of the selected techniques by [81]. From Table 5.4, we can observe that SWiT alongside randomly initialized WiT has a better performance compared to other works. Considering that our method can operate only on $N_c' = 32$ subcarriers and demands a reduced training duration, it offers a substantial advantage in computational efficiency. Furthermore, we outperform some localization techniques in small data regimes whilst training only a linear regressor on *frozen* parameters. For this evaluation setup, we use $R_{\text{test}} = 5\,000$.

Table 5.4: Localization error in [mm].

Method	$R = 1\,000$ ↓ MSE	$R = 5\,000$ ↓ MSE	$R = 10\,000$ ↓ MSE
Sun19 [49]	866.14	533.94	392.13
Arnold19 [38]	785.42	322.67	204.54
Chin20 [145]	660.77	248.25	118.75
Bast20 [34]	826.42	483.28	356.61
MHSA22 [146]	702.96	351.104	243.07
Wang18 [55]	1098.27	728.89	626.95
Hoang20 [39]	820.75	463.94	342.99
AAResCNN [81]	598.96	193.65	108.34
WiT	427.057	190.588	112.502
Ours:			
SWiT + Linear	570.42	467.6	459.08
SWiT + Fine-tuning	366.90	156.96	88.45

5.5.2 Transfer Learning

Throughout this dissertation, we have introduced models with the capability to achieve satisfactory localization accuracy in a specific environment. The motivation behind proposing SWiT, however, extends beyond such relatively straightforward approaches. We aim to investigate whether SWiT can learn more general-purpose channel features. To understand whether we learn useful macroscopic along with microscopic channel characteristics, we assess the transferability of the model across multiple datasets, environments, propagation conditions, and other wireless tasks. To do so, we use the pre-trained SWiT on the KUL-NLOS-Lab dataset and apply linear evaluation and fine-tuning on other datasets. Moreover, we apply learned representations to a relatively easy task, i.e., pathloss prediction. In this case, we compare it to the performance of transfer learning a fully-supervised model trained for wireless localization. Notwithstanding our goal to solve localization, this comparative analysis indicates that SWiT is a promising method to learn channel features that are useful for estimating other wireless communication tasks.

Transfer Learning to Other Datasets

We report the evaluated transferability of the pre-trained model to other datasets in Table 5.5, when $R = R_{\text{test}} = 5\,000$ samples. We can observe that despite noticeable, and sometimes even significant, variations in localization performance, overall, one can pre-train a model only on a single dataset while achieving relatively good localization performance across all environments and antenna configuration setups. For instance, a model trained only in the KUL-NLOS-URA dataset can achieve the same spot estimation performance on other KUL datasets and yields less than 2% degradation in `top-1` accuracy for S and HB datasets. Additionally, models trained on ray-tracing

channels (synthetic data) can sufficiently transfer learning to realistic environments. For example, the models trained on S and HB environments can achieve similar linear location estimates when evaluated on the KUL-LOS-DIS dataset.

Table 5.5: Transferability across different datasets for localization and *spot* estimation.

	KUL-NLOS-URA		KUL-LOS-ULA		KUL-LOS-DIS		S-200		HB-200	
Classifier:	Top – 1		Top – 1		Top – 1		Top – 1	Top – 5	Top – 1	Top – 5
KUL-NLOS-URA	**99.84**		99.98		99.82		16.1	44.34	20.68	53.58
KUL-LOS-ULA	98.86		**99.98**		99.43		17.12	46.16	17.8	49.18
KUL-LOS-DIS	99.10		99.98		**99.82**		13.04	40.14	20.4	56.14
S-200	99.12		99.96		99.52		**17.69**	**47.64**	12.1	35.74
HB-200	99.48		99.98		99.52		12.08	38.24	**21.05**	**56.14**
Linear Regression:	MAE	95−th	MAE	95−th	MAE	95−th	MAE	95−th	MAE	95−th
KUL-NLOS-URA	**453.894**	**908.275**	282.673	611.641	274.468	567.503	15.64	49.53	23.922	56.047
KUL-LOS-ULA	582.721	1098.16	**321.528**	**669.007**	286.267	590.466	16.702	49.18	27.82	67.53
KUL-LOS-DIS	557.30	1080.42	267.887	573.141	**281.402**	**596.043**	18.068	50.48	27.68	63.269
S-200	551.957	1068.88	278.177	584.959	273.491	565.191	**14.56**	**51.67**	37.24	78.422
HB-200	549.967	1065.52	273.01	582.07	277.331	569.638	18.69	50.140	**26.25**	**63.500**

Transfer Learning to Other Wireless Tasks

Next, we evaluate the transferability of supervised and self-supervised models to a task different from localization, i.e., pathloss prediction. In Table 5.6, we show the evaluation performance regarding the ability of a fully-supervised approach, trained on wireless localization, to transfer learning to pathloss prediction. This is denoted in Table 5.6 as 'Transfer-learning (Linear)'. As can be observed, the supervised model trained for channel-to-location mapping cannot transfer well to pathloss prediction. On the other hand, a model trained with a linear pathloss prediction head on top of the frozen features (i.e., 'SWiT+Linear' in Table 5.6), can achieve accuracy similar to the fully-supervised model trained on the same task (i.e., 'Fully-Supervised' in Table 5.6). For this evaluation, we split the whole S-200 dataset into 0.8 for training and 0.2 for evaluation and testing.

Table 5.6: Transfer Learning for path-loss prediction in [dB].

	S-200	
Method	↓ MAE	↓ 95−th
Fully-Supervised	5.917	18.493
Transfer-learning (Linear)	16.08	31.682
SWiT+Linear	6.594	18.426

5.5.3 Transformations and Fading Characteristics

We have previously reiterated the significance of the channel transformation function, or view selection, in facilitating the learning of invariant and robust channel features. Moreover, we also noted the importance of relying on exploiting small-scale channel features to improve the performance. In the following, we touch upon these two aspects of this work and examine different configuration setups to understand the impact better.

Impact of Channel Transformations

In Table 5.7, we show the results of an ablation study regarding the impact of channel transformations discussed in Sec. 5.4.1. We assess the performance by removing a transformation from the set $\mathcal{Q}$. More precisely, $\mathcal{T}_{\text{aug}}^{(1)}$ corresponds to the complete set of augmentations. When we use $\mathcal{T}_{\text{aug}}^{(2)}$, it matches the subset without *RSF*, $\mathcal{T}_{\text{aug}}^{(3)}$ indicates the subset without *flipping* and *RGO*. Further, when $\mathcal{T}_{\text{aug}}^{(4)}$, we add no *fading component*, i.e., we remove *RFC* too. Finally, for $\mathcal{T}_{\text{aug}}^{(5)}$, we use only *RSS* and *normalization*. In general, we observe that channel transformations slightly degrade the performance when assessed in KUL datasets for $R = R_{\text{test}} = 5\,000$. However, we observe improvements in the HB-200 scenario, and further gains can be emphasized when larger training dataset sizes are used during the fine-tuning. Overall, we can achieve high-quality self-supervised channel representations while unconstrained on the view selection function.

Relevance of Fading Characteristics

In Table 5.8, we show the relevance of learning macroscopic versus microscopic channel characterisitcs. To do so, we investigate the accuracy while varying $\beta = \{0.0001, 0.1, 0.3, 0.5, 0.7, 0.9\}$. We use $R = R_{\text{test}} = 5\,000$, and for the linear regression case, we train the linear head for 500 epochs. We can observe that relying on large-scale channel features is more relevant for the localization task, with the best

Table 5.7: Impact of channel transforms, $\mathcal{T}_{\text{aug}}(\cdot)$.

	KUL-NLOS-URA	HB-200		KUL-NLOS-URA		HB-200	
	Classification:			***Regression:***			
	Top − 1	Top − 1	Top − 5	MAE	95−th	MAE	95−th
$\mathcal{T}_{\text{aug}}^{(1)}$	99.64	21.12	56.16	445.2	901.0	23.22	57.05
$\mathcal{T}_{\text{aug}}^{(2)}$	99.83	18.66	53.24	446.537	908.679	23.92	58.90
$\mathcal{T}_{\text{aug}}^{(3)}$	99.76	19.46	52.9	435.319	864.965	23.60	58.66
$\mathcal{T}_{\text{aug}}^{(4)}$	99.72	20.18	54.06	429.302	857.742	23.54	57.26
$\mathcal{T}_{\text{aug}}^{(5)}$	99.96	20.08	53.94	420.821	833.4	24.11	57.38

performance achieved when the value of β lies in the range between 0.1 to 0.3. Performance degrades as we start to weight equally the loss for two learning tasks.

Table 5.8: Impact of β.

	KUL-NLOS-URA		
β	↑ Top-1	↓ MAE	↓ 95−th
0.0001	99.86	452.287	903.783
0.1	99.86	442.602	879.146
0.3	99.88	437.764	879.534
0.5	99.86	455.951	905.976
0.7	99.84	452.186	900.442
0.9	99.80	458.57	920.516

5.6 WiT Variants

In general, the main goal of WiT is to maintain and exploit the per-subcarrier channel structure through a transformer architecture. However, as the channel can also be represented over space and frequency, it is reasonable to explore alternative *slicing* strategies to feed the sequence of resulting slices (a.k.a tokens) into a transformer block. One motivation for investigating alternative *slicing* strategies is to incorporate the temporal dynamics of the environment into the learning process. By accounting for multiple geo-tagged channel estimates obtained over time from the same location and capturing dependencies between different tokens, we can improve the accuracy and robustness of the model. Further, we can take advantage of a prior knowledge on antenna array configuration. For instance, dividing the array into sub-arrays commonly helps to reduce the spatial correlation between the sub-arrays, which

benefits the learning of the underlying patterns between the resulting representations. Among multiple investigated designs, in this part, we show results for three most promising architecture choices which are illustrated in Fig. 5.8 and explained below. For the results that are depicted in Fig. 5.8, we train the models for 1 800 epochs on $R \approx 55\,000$ samples and report the normalized validation error (NMSE) in dB. We note that different models demand varying input information, leading to distinct training sample sizes; hence, cross-comparison may be considered unfair.

WiT-E-TF

Here, we aim to exploit the temporal variations of the environment by stacking T channel matrices obtained from the same location and viewing the channel as $\mathbf{H}_r \in \mathbb{C}^{N_r \times N_c' \times T}$. Despite stacking multiple channel matrices to leverage temporal channel realizations, the fundamental input channel representation to the design remains $\mathbf{h}_i \in \mathbb{R}^{1\times 3N_r}$. Again, this representation denotes a single-time snapshot. However, the computational burden increases as we aim to utilize the temporal variations by considering T snapshots. Specifically, considering T representations would require $C_{\text{wit}} = (N_c'T+1)$ computations, instead of $C_{\text{wit}} = (N_c'+1)$ we defined in self-attention in Sec. (5.1.1). To address this issue, we propose to employ a factorized dual-attention approach, resulting in a reduction of computational complexity to $(N_c'+T+2)$ per input channel vector representation. This is achieved by considering and learning separate sets of weights, $\{\mathbf{W}_q^{(1)}, \mathbf{W}_k^{(1)}, \mathbf{W}_v^{(1)}\}$ (denoted as 'Attention - Time Samples' in Fig. 5.8a), for the first attention, and $\{\mathbf{W}_q^{(2)}, \mathbf{W}_k^{(2)}, \mathbf{W}_v^{(2)}\}$ (i.e., 'Attention - Subcarriers' in Fig. 5.8a) for the second attention within a transformer block. The attention is first evaluated for the $i-$th subcarrier over T channel realizations, resulting in an embedding $\widetilde{\mathbf{o}}_i$. Subsequently, to obtain $\mathbf{o}_i$, we pass $\text{LayerNorm}(\widetilde{\mathbf{o}}_i + \hat{\mathbf{e}}_i; \zeta, \iota)$ to the second attention. Attention is evaluated as in equation (5.2). In Fig. 5.8b, we show the performance while varying $T = \{4, 8, 12, 16\}$, and keeping the number of subcarriers constant, $N_c' = 32$. While the localization accuracy improves compared to $T = 8$ (i.e., S32/T8), it starts to degrade for $T > 8$. However, we note that as we increase T, the available training dataset size is reduced too, hence, resulting in unjust comparison. Nevertheless, we consider S32/T8 model as a favorable tradeoff between the computational complexity and the performance.

WiT-E-AF

In contrast to the aforementioned time-frequency separated attention approach, here we view the channel as $\mathbf{H}_r \in \mathbb{C}^{N_c' \times 1 \times N_r}$, and we further *slice* subcarriers into S parts for each antenna element, resulting in an input representation vector $\mathbf{h}_i \in \mathbb{R}^{1\times 3N_c'/S}$. We depict antenna-frequency separated attention in Fig. 5.8c, where $\widetilde{\mathbf{o}}_i$ and $\mathbf{o}_i$ are obtained similar to that described in WiT-E-TF. The localization performance is shown in Fig. 5.8d for $N_r = \{4, 8, 16\}$, and $S = 8$. This design choice performs poorly

for co-located antennas, whilst gains are clearer in distributed antenna systems as $N_r > 4$.

WiT-E-UPA

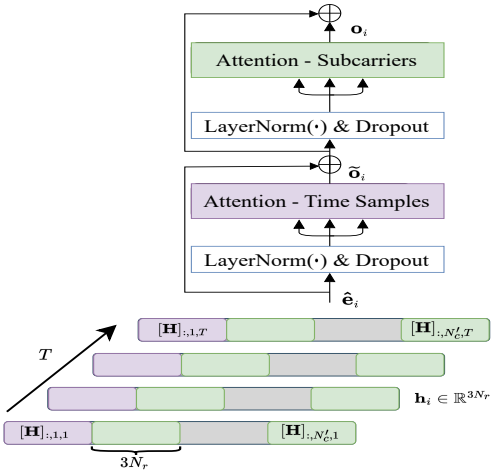
We also attempt to leverage the prior knowledge on the alignment of antenna elements in an array structure or clusters of RRHs. For the co-located antennas, we do so by viewing the channel as $\mathbf{H}_r \in \mathbb{C}^{N_{r_z} \times N_{r_x} \times N'_c}$, where N_{r_z} denotes the number of antenna elements aligned along the height of the array and N_{r_x} along its width. Further, we consider $P = P_1 P_2$ sub-arrays, where P_1 and P_2 are the *slicing* factors across height and width of the array, respectively. The flattened patch for the $i-$th subcarrier results in an input representation vectors $\mathbf{h}_i \in \mathbb{R}^{1 \times 3 \frac{N_{r_z}}{P_1} \frac{N_{r_x}}{P_2}}$. We depict UPA attention design in Fig. 5.8e. Similar to the above elaborated designs, we first evaluate the attention for the channel vector at each sub-array over all subcarriers, then process the *attended* representation through second attention to evaluate over other sub-arrays. For DAS, we consider P different clusters each having M/P adjoint RRHs, i.e., RRHs within a particular cluster are in closer physical distance compared to those in other clusters. The performance is shown in Fig. 5.8f for $P = \{2, 4, 8\}$, and $S = 8$. As we can observe, the performance improves in a co-located scenario with $P > 2$, but it degrades with increasing number of clusters of RRHs in the case of DAS.

5.6.1 Number of Parameters

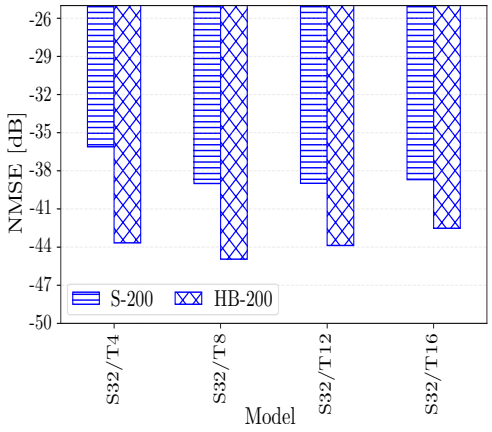
Table 5.9: Input representation and the number of parameters.

	S-200		
Model	↓ MAE	↓ 95−th	↓ Par. ($\times 10^6$)
WiT-E-TF S32/T8	1.46	4.29	4.78
WiT-E-AF S8/A16	4.33	13.01	4.66
WiT-E-UPA S8/P8	1.79	4.45	4.67
WiT [LID] [87]	2.36	6.54	2.67
WiT-L [LID]	**1.45**	**3.78**	**4.41**
SWiT+Fine-tuning	1.98	5.66	1.86

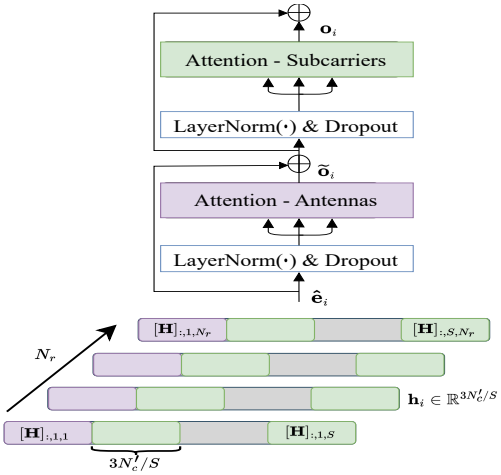
In our work, we set up single-headed transformer blocks to trade-off between model complexity and expressivity. However, we acknowledge that overparameterization could improve the model's performance considerably. In Table 5.9, we compare the performance among the proposed approaches in this work for the best performing



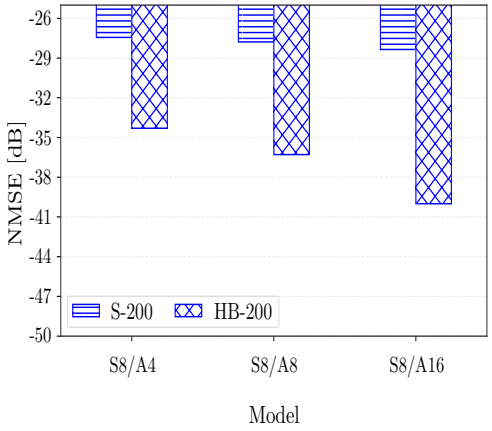
(a) Time-frequency (WiT-E-TF).



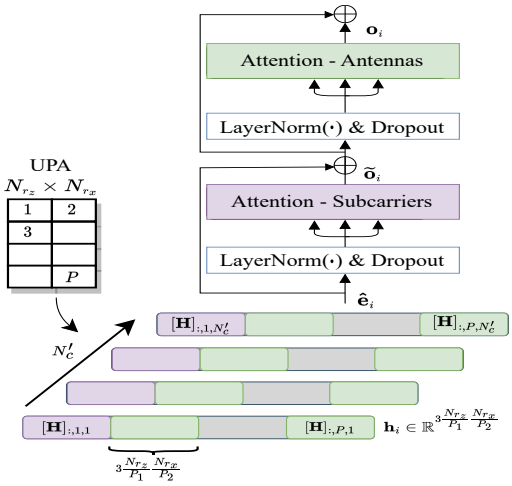
(b) WiT-E-TF.



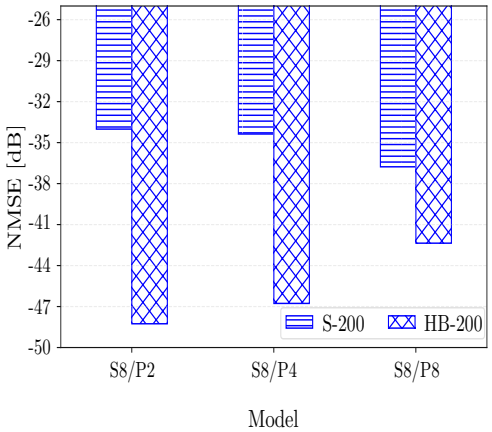
(c) Antenna-frequency (WiT-E-AF).



(d) WiT-E-AF.



(e) Frequency-antenna (WiT-E-UPA).



(f) WiT-E-UPA.

Figure 5.8: WiT variants and achievable performance.

models. Furthermore, we also show the number of trainable parameters for each model, with only the parameters for the encoder and linear MLP head shown for 'SWiT+Fine-tuning'. Considering the results from the above discussed investigation, we show that incorporating time information improves localization performance for both co-located and DAS. Nevertheless, this increases the number of trainable parameters compared to WiT. Thus, to understand the impact of computational complexity, we increase the depth of the WiT (denoted by 'WiT-L' in the Table 5.9), i.e., increase the number of transformer blocks to $H_{\text{blcn}} = 3$, each having $H_{\text{attn}} = 6$ and $D = 128$. By doing so, the number of parameters becomes comparable to WiT-E models, and the accuracy improves (even surpasses them). Moreover, in Table 5.9, we also compare the self-supervised approach proposed in this work (i.e., 'SWiT+Fine-tuning'). We see that it has lower computational complexity and performs on par with fully-supervised approaches, and even better in some instances. Finally, in Table 5.10, we provide additional information useful to quantify the time consumption. In addition to the inference and training *throughput*, measured in iterations per second, we have also included an estimate of a more hardware-independent metric, such as the number of floating points (FLOPs). For SWiT, we calculate the FLOPs during the propagation of a global channel view (i.e., 'SWiT - global view' in Table 5.10). SWiT alone is not utilized during the testing phase; hence, no value is shown in the respective column of Table 5.10. In the case of the linear approach (i.e., 'SWiT+Linear'), we evaluate operations for the linear head. We note that the computational overhead of deploying and fine-tuning SWiT at the base station (or central unit) can be significant and lead to poor real-time performance. Therefore, for applications in mobile networks, we seek to further enhance the model's performance only with a linear head, which may hold more practical value.

Table 5.10: Time complexity of SWiT.

	↓ FLOPs	↑ Training [iter/sec]	↑ Testing [iter/sec]
SWiT - global view	1.28×10^9	1.13	-
SWiT + Fine-tuning	0.06×10^9	16.02	443.66
SWiT + Linear	768	456.21	1183.60

5.7 Final Remarks

In this chapter, we first addressed the issue related to the sensitivity of CSI (e.g., to small variations in the surroundings), which poses a significant challenge for basic DNN architectures, such as those relying on MLPs, in learning useful channel features for wireless localization. Hence, we proposed a transformer-based model, WiT. WiT

demonstrates learning more robust channel features and, consequently, shows better localization performance, even in scenarios dominated by strong multipath. We then extended WiT and proposed SWiT, a joint-embedding self-supervised method for wireless channel representation learning. We crafted SWiT to learn invariant and compressed channel representations from macroscopic and microscopic fading channel characteristics. We showed that our proposed framework could outperform other state-of-the-work localization techniques in small-data regimes, even under a linear evaluation approach. Additionally, we discussed how the pre-trained model on one dataset could be used to assess performance for *spot* estimation in other datasets from different environments or system configurations. Moreover, we found that the learned representations are transferable to other wireless downstream tasks. For instance, we were able to achieve highly accurate pathloss predictions using only a linear MLP head.

Furthermore, we proposed multiple variants of transformer-based models and evaluation approaches. Consequently, an essential consideration emerges regarding the optimal selection of a particular variant based on specific application requirements. We showed that linear models (i.e., single linear MLP heads) demonstrate similar localization accuracy as other much more complex methods in scenarios with minimal data. Such low-complexity models trained on smaller datasets can be beneficial in scenarios where centralized processing poses challenges, like latency and communication overhead. Hence, it is desirable to enhance the performance of models with a single linear head to remain competitive even with larger datasets. When sufficient training data is available and computational complexity is not a constraint in situations demanding higher performance, one might consider various WiT flavors trained in a fully supervised setting or fine-tune the encoder based on available information and desired localization accuracy.

The transformer-based methods proposed in this chapter are specifically tailored for OFDM-based wireless systems, including present-day cellular and WiFi. However, the transferability of the representations to other wireless systems or waveforms remains a challenge. In principle, applying the method, e.g., on top of the delay-Doppler channel representation relevant for orthogonal time frequency space (OTFS) modulation [147], should also be possible. Nevertheless, potential issues with generalizing learned representations may arise due to variable subcarrier spacing settings in standardized 5G and 6G mobile networks.

6

Conclusion and Outlook

Wireless networks are predominantly acknowledged for their communication capabilities, with the added potential for localization and sensing increasingly recognized recently by the research community. Historically treated as distinct from mainstream communication research, wireless localization is now being considered among the most critical tasks for the next generation of mobile networks. As we progress towards 6G, the increase in antennas at the BS and the shift to higher frequencies is expected to play a crucial role in further improving the accuracy and reliability of advanced wireless networks for localization. Concurrently, integrating AI approaches will improve mobile network services and seamlessly merge communication and sensing capabilities.

Achieving precise localization from raw channel measurements, e.g., CSI in massive MIMO systems, necessitates advanced techniques that go beyond traditional models and signal processing techniques. This is particularly true when dealing with multipath channels and noisy measurements or when the system's nonlinear signal characteristics are either unknown or intractable. Therefore, data-driven approaches are a promising direction for modeling the complex, non-obvious relationships between the received signal information and the UE position.

The main contribution of this dissertation is the development and investigation of advanced deep learning algorithms for wireless localization, with a specific focus on uplink CSI acquired at a massive MIMO system. Below, we summarize the main findings and contributions of this dissertation.

6.1 Summary of Contributions

The first part of the dissertation focused on supervised learning for wireless localization. In Chapter 3, we addressed the challenge of measuring the confidence of estimates obtained by existing DNN-based wireless localization methods. We proposed and investigated two variational DNN approaches for quantifying the uncertainty of location estimates in a co-located massive MIMO system. We showed that modeling the DNN output as a mixture of Gaussians is suitable to acquire the uncertainty due to the changes in propagation conditions. Applying MC with dropout as a stochastic regularization technique during inference or utilizing an ensemble approach improves the ability of the DNN model to identify out-of-set regions, i.e., regions from where the model has not previously encountered any training data. We found that an ensemble of trained DNN models significantly improves the localization performance compared to solely utilizing the stochastic dropout approach.

In Chapter 3, we also investigated the localization performance in a DAS. We showed that increasing the number of RRHs improves the overall performance but can also compensate for the impact of the dominant NLOS components. We then extended the uncertainty-aware DNN method to incorporate a learning strategy that can be employed to identify and select a subset of the most relevant RRHs. We demonstrated that while the selection strategy results in performance degradation compared to using the complete set of RRHs, it still outperforms a heuristic approach based solely on maximum channel gain.

The second and third parts of the dissertation were dedicated to learning compressed channel features. In Chapter 4, we transitioned from supervised to unsupervised learning. To exemplify the localization performance of shallow feature extractors, we investigated a linear subspace and a manifold learning approach. Subsequently, we introduced a metric-learning DNN method for learning low-dimensional wireless channel features. We introduced a contrastive task to train a Siamese-based DNN that can discriminate wireless channel embeddings across different locations while grouping those from similar ones. We then demonstrated that a four-dimensional channel representation is satisfactory for capturing the relevant channel characteristics and revealing a unique UE spatial signature. The metric-learning approach we elaborated on in this chapter demonstrated good performance gains even in small data regimes and when only NLOS signal paths exist.

In Chapter 5, we introduced a transformer-based method and a non-contrastive self-supervised approach for learning channel representations. We showed that the transformer-based model learns more robust channel features and performs better localization compared to a base DNN, even in scenarios dominated by strong multipaths. We then crafted a non-contrastive approach to learn invariant and compressed channel representations from macroscopic and microscopic fading channel character-

istics. We also revealed that we can outperform other state-of-the-work localization techniques in small-data regimes and NLOS conditions, sometimes even with a model composed of a single-linear layer. We found that the learned representations are transferable to other wireless downstream tasks beyond localization. Finally, we investigated multiple variants of transformer-based models and evaluation approaches. We showed that incorporating channel measurements at different times improves localization performance for both co-located massive MIMO and a DAS at the cost of computational complexity.

6.2 Open Issues and Possible Future Works

Channel models and datasets ML algorithms rely on scenario-specific datasets to acquire the statistical knowledge required for making predictions. Similar to other fields where there exist standardized datasets, such as UCI [148], comprehensive and standardized channel measurements have become necessary for validating wireless localization performance of different models in a variety of scenarios and propagation settings. In this dissertation, we relied on either actual measurements taken from real-world environments or synthetic data. However, the need within the research community for a unified and standardized training and testing datasets is essential for enabling fair comparisons and tracking progress in wireless localization research.

Achieving high localization accuracy in specific datasets is just one aspect of the ML research; comprehending how different propagation conditions, channel estimation error, and hardware impairments (e.g., phase noise and mutual coupling) affect performance is equally important. For instance, in this dissertation, we utilized established channel models like the geometric model outlined in Chapter 2, Sec. 2.1.2 to generate synthetic data in simulated scenarios. This approach helped us examine the influence of LOS availability. Furthermore, in Chapter 3, Sec. 3.3, a shift to an alternative channel model was necessary to assess the impact of NLOS strength on localization accuracy. Consequently, it is crucial to have advanced simulators, such as [149, 150], to interface with ML libraries and be less computationally demanding. This would facilitate more extensive simulations of industry-standard communication systems, enabling a better understanding of the limitations and practical application of the algorithms proposed in this dissertation.

Performance bounds and evaluation approaches For model-based wireless localization methods, it is common to derive fundamental performance bounds, such as those based on Fisher information or the Cramér-Rao bounds, which offer theoretical limits on the accuracy of the estimators. However, deriving such bounds for DNN-based methods presents a challenge, primarily due to the intractable complexity and the lack of explicit mathematical formulations that relate inputs to outputs.

In Chapter 3, Sec. 3.2, we introduced the data and model ambiguity, suggesting that measures of uncertainty can serve as effective heuristics for understanding performance limitations under different system configuration setups. However, such measures are very sensitive to hyperparameter choices. A notable example in this chapter, is the need for meticulous tuning of dropout rates in stochastic dropout methods, which significantly influences the reliability of confidence measures derived from the model. Hence, developing more principled approaches and formal methods for evaluating and monitoring the performance of DNN methods is necessary.

Similarly, in Chapter 5, we proposed an evaluation methodology for SSL models, a new research direction in wireless communications, specifically in wireless localization. As SSL gains more attention in our field, motivated by its success in NLP and computer vision, it is essential to maintain the linear evaluation approaches to assess the quality of learned channel features for foundation models. Furthermore, it is important to distinguish between SSL, particularly non-contrastive learning methods, and other approaches like meta-learning, contrastive metric-learning, or autoregressive models. Such clarity will benefit the development of foundational models that can be applied to various wireless tasks, and aligning the DNN-based wireless research with the broader advancements in the machine learning community. Therefore, similar to [151], a comprehensive yet specific survey of SSL in wireless communications can be valuable.

Channel transformations and transferability To train the SSL method presented in Chapter 5, we developed pretext tasks and channel transformations, resulting in useful channel embeddings for wireless localization. However, the effectiveness of the proposed transformations in enhancing robustness and the quality of channel representations may vary across different datasets and lacks formal motivations. Therefore, a future research direction involves a more rigorous investigation into the impact of these channel transformations, particularly across diverse datasets. While there have been some efforts in the direction of theoretical understanding of the learning dynamics of non-contrastive SSL methods, such as [152, 153], their findings do not necessarily extend to different optimization objectives, including varied loss functions and pretext tasks, nor do they directly apply to the wireless signal information used in this dissertation.

The potential for issues in generalizing learned representations arises with the variable subcarrier spacing in 5G and 6G networks. In Chapter 5, we also showed that obtained channel representations can be used for other wireless downstream tasks, such as pathloss prediction. An interesting future work is investigating whether these representations can be useful for CSI feedback. Hence, a more comprehensive study is necessary to validate the generalization capabilities in varied subcarrier spacing, other wireless tasks, and waveforms.

Computational complexity The computational overhead of deploying and fine-tuning the transformer-based models at the BS can be significant. In Chapter 5, we showed that single-head linear models demonstrate localization accuracy similar to other, much more complex methods in scenarios with minimal data. In the future, we desire to enhance the performance of such models to remain competitive even with larger datasets, which may hold more practical value for applications in mobile networks.

Codebase and reproducibility To facilitate the reproducibility and benchmarking of various models proposed in this dissertation, we plan to unify and compile the codebase into a software package. This will ensure that researchers and practitioners can easily access, experiment with, and build upon the proposed methods. The codebase will be made publicly available [1]. Currently, the logs and separate codebases are accessible in a shared folder, which can be found at [2].

[1] https://github.com/ars205/nextlocai

[2] https://drive.google.com/drive/folders/1KmfHXVbzOAwwilCLdJZrIjHOVZDPX1ej?usp=drive_link

A

System Model

A.1 Notations and Symbols in Chapter 2

Table A.1: A short description of the most relevant symbols.

Symbol	Description
N_r	Number of antenna elements at the base station
M	Number of remote radio heads (RRHs)
$\mathbf{b}_m$	Position of m-th RRH
R	Number of single-antenna user locations
$\mathbf{u}_r$	Position of r-th user location
G	Number of scattering objects in the region of interest (ROI)
$\mathbf{p}_g$	Position of g-th scattering object
P	Average transmit power
x_n	Normalized transmitted signal on subcarrier n
$\tilde{\mathbf{h}}_n$	Channel vector on subcarrier n
$\mathbf{n}_n$	Noise vector on subcarrier n
N_0	Noise power spectral density
$\mathbf{h}_n$	Estimated channel state information on subcarrier n
L_{path}	Number of propagation paths
η_ℓ	Complex gain of ℓ-th path
τ_ℓ	Propagation delay (ToA) of ℓ-th path
$\varphi_{az,\ell}$	Angle of arrival (AoA) in azimuth of ℓ-th path
$\varphi_{el,\ell}$	Angle of arrival (AoA) in elevation of ℓ-th path
Δf	Subcarrier spacing
$\mathbf{a}_x$	Array steering vector along x-axis
$\mathbf{a}_z$	Array steering vector along z-axis
λ_c	Wavelength of carrier frequency
f_c	Carrier frequency
c	Speed of light
d	Antenna element spacing
T	Time
$\mathring{w}_{g,i}$	Zero-mean Gaussian noise term with variance $\sigma^2_{\mathring{w}}$
$\dot{w}_{g,i}$	Zero-mean Gaussian noise with variance $\sigma^2_{\dot{w}}$
$\mathbf{H}_r$	Matrix of subcarrier vectors for r-th user location
$\boldsymbol{\Theta}$	Parameters of the neural network
$J(\boldsymbol{\Theta})$	Cost function of the neural network
$\mathcal{L}(\cdot)$	Loss function of the neural network
$f_{\boldsymbol{\Theta}}(\mathbf{H}_r)$	Mapping function of the neural network

B

Towards Dependable Location Estimates

B.1 Indoor Scenario in Chapter 3

Below is the illustration of indoor scenario considered for evaluating the LUD for indoor environments. Similar to the outdoor scenario, UEs can be in LOS or NLOS.

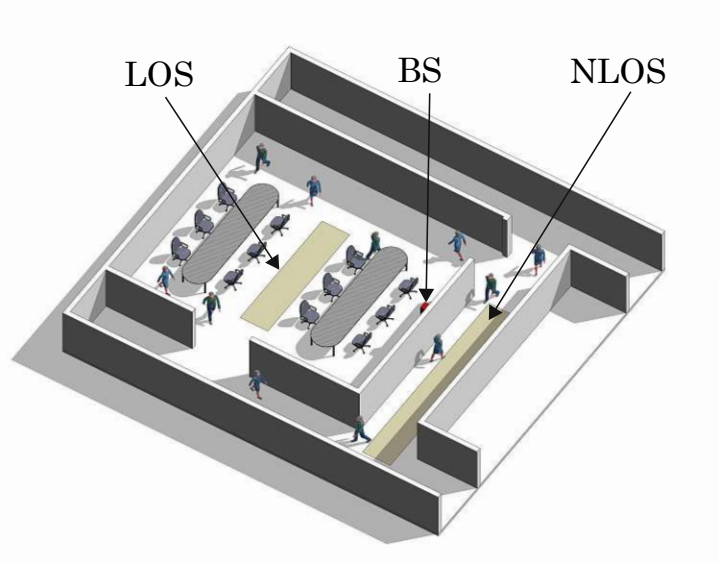


Figure B.1: Indoor scenario considered in Chapter 3.

B.2 Notations and Symbols in Chapter 3

Table B.1: A short description of the most relevant symbols.

Symbol	Description
L	Number of hidden layers
M	Number of RRHs
$\bar{\mathbf{h}}_r$	Real-valued input channel to DNN
$\mathcal{D}$	Dataset with $(\bar{\mathbf{h}}_r, \mathbf{u}_r)$
$\boldsymbol{\Psi}_b$	Parameters the base DNN
$\boldsymbol{\Psi}$	Parameter for LUD
C_{mix}	Total mixtures
$\omega_{\boldsymbol{\Psi},c}$	Mixture weight fo c-th Gaussian
$\boldsymbol{\mu}_{\boldsymbol{\Psi},c}$	Mean for c-th Gaussian
$\boldsymbol{\sigma}^2_{\boldsymbol{\Psi},c}$	Variance for c-th Gaussian
$o^{\omega}_{\boldsymbol{\Psi},c}$	Input to last mixture layer
$\widetilde{\boldsymbol{\mu}}_{\boldsymbol{\Psi}}$	Final mean
$\widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}}$	Final variance
$\widetilde{\boldsymbol{\sigma}}^2_{\boldsymbol{\Psi}}$	Data uncertainty
S_{mc}	Forward stochastic passes or ensemble
Φ_{drop}	Dropout rate
α_{R_i}	Error between oracle RMSE and confidence RMSE
b_{R_i}	Fraction of removed locations
$\alpha_{r,m}$	LOS path coefficient between r and m
$h_{r,m}$	Channel coefficient between r and m
$\beta_{r,g}$	Scattering coefficient between r and g
φ_j	Random phase-shift at scattering object
γ	Scattering coefficient values
K_{rsd}	Number of selected RRHs
$\boldsymbol{\Theta}$	Parameters of RSD
$\widehat{\mathbf{h}}_r$	CSI from RRHs (or input to LUD which Stefan says does not make sense)
$\mathbf{A}$	Selection matrix
z_k	Categorical variable
$\boldsymbol{\pi}_k$	Probability scores for z_k
$\boldsymbol{g}_k$	Gumbel noise
$\boldsymbol{\phi}_k$	Parameters in selection layer
$\tau_{t_{\text{epoch}}}$	Temperature value at epoch

B.3 Importance Sampling for Active Learning

In this section, we illustrate the application of model uncertainty in importance sampling (IS) for fine-tuning LUD with a deep ensemble network (DEN) discussed in Chapter 3. First, let us denote by $\mathcal{D}_{\text{ini}} = \{(\mathbf{h}_i, \mathbf{u}_i)\}_{i=1}^{R_{\text{ini}}}$ the initial training dataset comprising channel estimates and corresponding location labels. We also define an unlabeled data pool of channel estimates as $\mathcal{D}_{\text{unl}} = \{\mathbf{h}_j\}_{j=1}^{R_{\text{unl}}}$. After the initial training, the ensemble model evaluates $\mathcal{D}_{\text{unl}}$ to compute the uncertainty measure $\widehat{\boldsymbol{\sigma}}^2_{\text{model}}$ for each sample. Then, we apply the query strategy $\mathcal{Q}_{\text{IS}}$, that selects P_{IS} samples with the highest position uncertainty output values. These samples are labeled with their true locations to form a new dataset $\mathcal{D}_{\text{IS}} = \{(\mathbf{h}_{r'}, \mathbf{u}_{r'})\}_{r'=1}^{P_{\text{IS}}}$. Moreover, we removed the selected samples from the unlabeled data pool to ensure they are not reselected in future iterations, $\mathcal{D}_{\text{unl}} \leftarrow \mathcal{D}_{\text{unl}} \setminus \mathcal{D}_{\text{IS}}$. Finally, we create a combined dataset $\mathcal{D}_{\text{new}} = \mathcal{D}_{\text{ini}} \cup \mathcal{D}_{\text{IS}}$, and fine-tune DEN.

In Fig. B.2, we show the impact of IS for training DEN with $R_{\text{ini}} = 25\,000$ and $P_{\text{IS}} = 20$ in an active learning loop for O1_3p5B dataset [110]. Compared to a random sampling strategy (RAND), DEN, with the uncertainty-based query strategy explained above, shows consistent improvements.

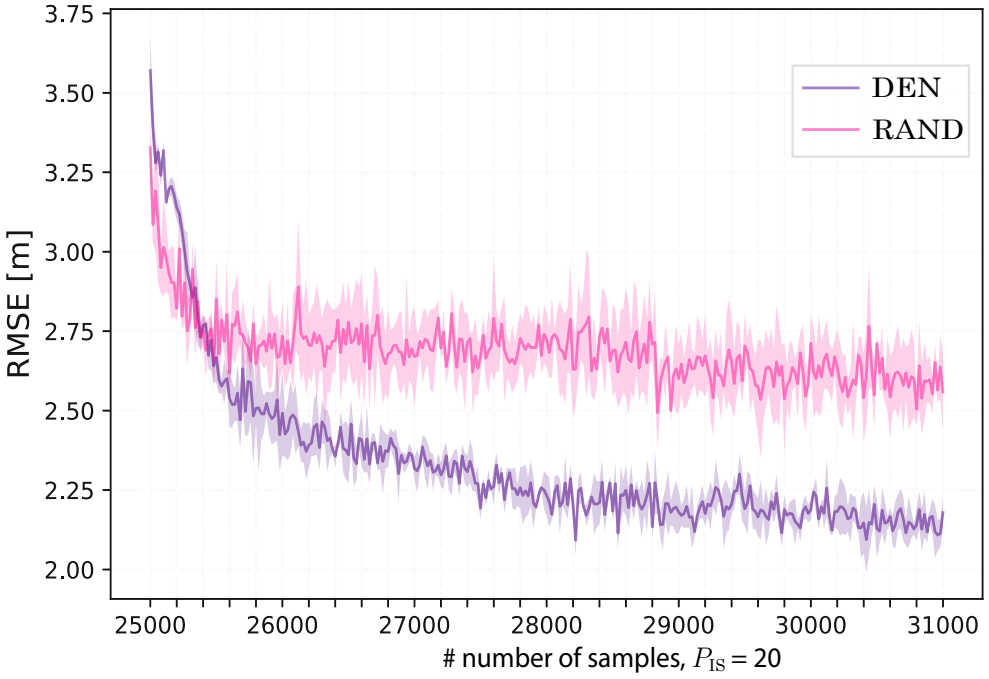


Figure B.2: Importance sampling for active learning.

C

Shallow and Deep-Metric Learning

C.1 Notations and Symbols in Chapter 4

Table C.1: A short description of the most relevant symbols.

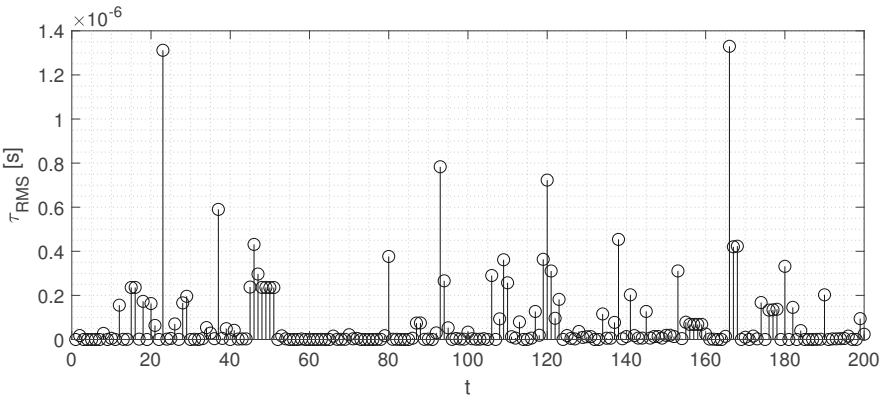
Symbol	Description
$\mathbf{Z}_{\text{PCA}}$	Low-dimensional map for PCA
σ_i	Singular values associated with PCA
$\mathbf{u}_{\text{PCA},i}$	Eigenvectors corresponding to the singular values in case of PCA
$\mathbf{C}_{\text{dis}}$	Distance matrix used in MDS
$\mathbf{B}_{\text{dis}}$	Matrix obtained by double-centering $\mathbf{C}_{\text{dis}}$ in MDS
$\mathbf{Z}_{\text{MDS}}$	Low-dimensional map for MDS with Sammon mapping
$\mathcal{R}_{\text{rps}}$	Set of reference point (RP) locations
$d_{\mathbf{z}_r}$	Distance metric in the Z-map to each RP location
r_{NN}	Nearest RP based on the minimum Eucledian distance
λ_r	User density value used in partitioning schemes
R_{rps}	Number of sub-regions for triplet mining
K_{rps}	RPs with highest probability score
$\bar{\mathbf{h}}_i^{(p)}$	Positive sample in the triplet network
$\bar{\mathbf{h}}_i^{(n)}$	Negative sample in the triplet network
R_{tri}	Number of triplets for training in the triplet network
$\mathcal{L}_{\text{tri}}$	Triplet loss function in the Siamese-based approach
δ_{tri}	Triplet margin used in the loss function
$\mathbf{z}_i^{(a)}, \mathbf{z}_i^{(p)}, \mathbf{z}_i^{(n)}$	Embedding of the anchor, positive, and negative sample

D

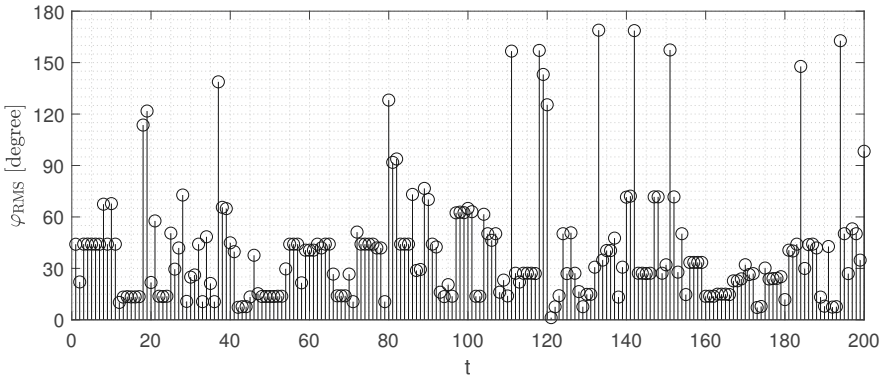
Self-Supervised Representations

D.1 S-scenario discussed in Chapter 5

Below we plot the RMS delay-spread, τ_{RMS}, as well as the RMS angle of arrival spread in azimuth, φ_{RMS}, for a single random UE location over $T = 200$ time snapshots and $L_{\text{path}} = 4$ strongest paths.



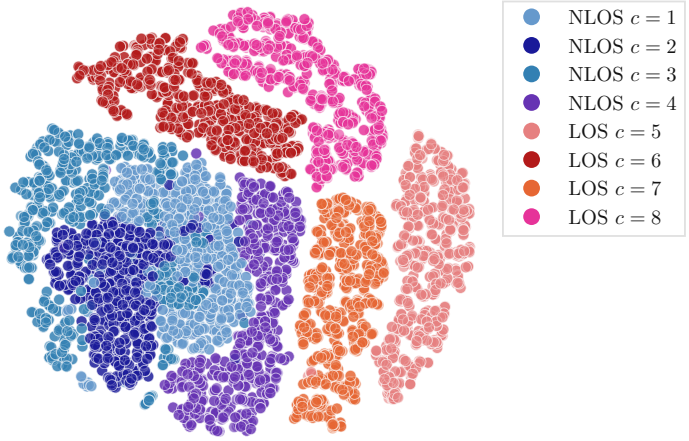
(a) An example of RMS delay spread.



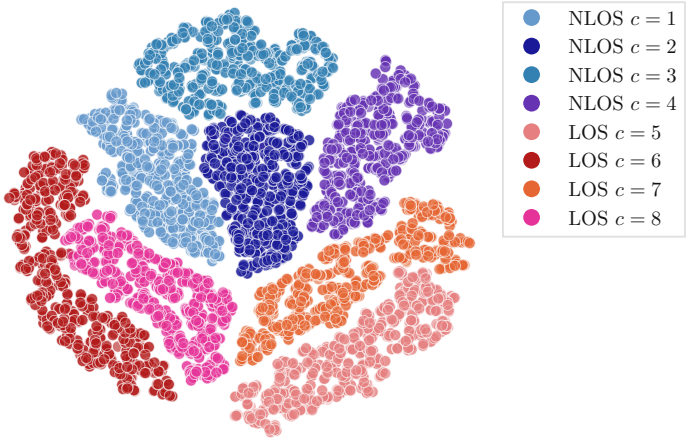
(b) An example of RMS angle of arrival spread in azimuth

Figure D.1: Delay- and angle-spread at $\mathbf{u}_r$ over $T = 200$ time snapshots.

D.2 t-SNE Embeddings for Combined Dataset



(a) Random.



(b) SWiT.

Figure D.2: Two-dimensional t-SNE embeddings for a combined dataset.

D.3 Notations and Symbols in Chapter 5

Table D.1: A short description of the most relevant symbols.

Symbol	Description
$f_{\boldsymbol{\Theta}}(\cdot)$ and $f_{\boldsymbol{\Psi}}(\cdot)$	Online and target encoders.
$\mathcal{L}_{\text{SSL}}(\cdot)$	SSL loss function.
$g_{\boldsymbol{\Phi}}(\cdot)$	MLP head.
$\mathbf{g}_i$	Positional encoding.
$\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v$	Attention weights.
$\mathbf{o}_i$	Embedding after the attention operation.
$\alpha_{i,j}$	Attention coefficient.
$\hat{\mathbf{e}}_i$	Input to the transformer block.
ζ and ι	Layer norm hyperparameters.
κ_{per}	Types of materials for RT.
$\bar{\mathbf{o}}_i$	Subcarrier output (target encoder or WiT).
$\mathcal{Q}$	Set of stochastic transformations.
$\mathcal{T}_i(\cdot)$	Channel transformation function.
$\bar{\mathbf{h}}'_n, \tilde{\mathbf{h}}'_n, \hat{\mathbf{h}}'_n$	Transformed subcarrier views.
N'_{c_i}	Selected subcarriers for respective transformations.
V	Number of channel views.
D	Dimensionality of $\bar{\mathbf{o}}_i$ (or $\bar{\bar{\mathbf{o}}}_i$).
C	Total subcarrier embeddings (including $\bar{\mathbf{o}}_0$).
γ_{c_i}	Ratio factor of selected adjacent subcarriers.
$N'_{\text{cg}}, N'_{\text{cl}}$	Embeddings to encoders for global and local views.
P_{flip}	Permutation (flipping) channel matrix.
N_s	Number of local views.
ξ_{rgo}	Random gain offset.
σ_{rfc}	Fading component scaling factor.
$\omega_{n,i}, \sigma_Q^2$	Gaussian noise hyperparameters.
$\tilde{\mathbf{z}}_0^{(v)}, \tilde{\tilde{\mathbf{z}}}_0^{(v)}, \bar{\mathbf{z}}_0^{(v)}, \bar{\mathbf{z}}_{\text{NN}_i}^{(v)}$	Final channel representations from the projectors.
$\nu(\cdot)$	Attention pooling operation.
$\chi_{(\cdot)}$	Temperature scaling for target or online encoders.
$\boldsymbol{\epsilon}$	Centering of the output embedding.
K_n	Subset of nearby subcarriers from $\mathcal{N}_i$.
$\rho(\cdot,\cdot)$	Correlation of nearby subcarriers.
β	Loss function scaling hyperapameter.
B	Batch size.
ϖ_{base}	Base learning rate.
P	Number of sub-arrays.

Bibliography

[1] Ericsson, "Ericsson mobility report," Ericsson: Stockholm, Sweden, Jun 2023.

[2] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6G wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166–1199, 2021.

[3] B. Rong, "6G: The next horizon: From connected people and things to connected intelligence," *IEEE Wireless Communications*, vol. 28, no. 5, pp. 8–8, 2021.

[4] X. You, C.-X. Wang, J. Huang, X. Gao, Z. Zhang, M. Wang, Y. Huang, C. Zhang, Y. Jiang, J. Wang *et al.*, "Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts," *Science China Information Sciences*, vol. 64, pp. 1–74, 2021.

[5] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle, *GNSS–global navigation satellite systems: GPS, GLONASS, Galileo, and more*. Springer Science & Business Media, 2007.

[6] P. D. Groves, "Principles of GNSS, inertial, and multisensor integrated navigation systems, [book review]," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, no. 2, pp. 26–27, 2015.

[7] E. D. Kaplan and C. Hegarty, *Understanding GPS/GNSS: Principles and applications*. Artech house, 2017.

[8] G. Seco-Granados, J. Lopez-Salcedo, D. Jimenez-Banos, and G. Lopez-Risueno, "Challenges in indoor global navigation satellite systems: Unveiling its core features in signal processing," *IEEE Signal Processing Magazine*, vol. 29, no. 2, pp. 108–131, 2012.

[9] T. Li, H. Zhang, Z. Gao, Q. Chen, and X. Niu, "High-accuracy positioning in urban environments using single-frequency multi-GNSS RTK/MEMS-IMU integration," *Remote sensing*, vol. 10, no. 2, p. 205, 2018.

[10] J. A. del Peral-Rosado, R. Raulefs, J. A. López-Salcedo, and G. Seco-Granados, "Survey of cellular mobile radio localization methods: From 1G to 5G," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1124–1148, 2017.

[11] 3GPP, "Location services (LCS); functional description; stage 2, release 98, v7.0.0," 3GPP, Sophia Antipolis, France, Rep. 3GPP TS 03.71, Jun 1999.

[12] E. T. S. Institute, "Digital cellular telecommunications system (phase 2+); location services (LCS); service description, stage 1 (gsm 02.71 version 7.0.0 release 1998)," ETSI, Technical Specification ETSI TS 101 723 V7.0.0, Aug 1999.

[13] J. Medbo, I. Siomina, A. Kangas, and J. Furuskog, "Propagation channel impact on LTE positioning accuracy: A study based on real measurements of observed time difference of arrival," in *2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2009, pp. 2213–2217.

[14] T. Wigren, "LTE fingerprinting localization with altitude," in *2012 IEEE Vehicular Technology Conference (VTC Fall)*. IEEE, 2012, pp. 1–5.

[15] X. Lin, J. Bergman, F. Gunnarsson, O. Liberg, S. M. Razavi, H. S. Razaghi, H. Rydn, and Y. Sui, "Positioning for the internet of things: A 3GPP perspective," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 179–185, 2017.

[16] J. Blumenstein, A. Prokes, T. Mikulasek, R. Marsalek, T. Zemen, and C. Mecklenbräuker, "Measurements of ultra wide band in-vehicle channel-statistical description and TOA positioning feasibility study," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, pp. 1–10, 2015.

[17] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, "A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 829–846, 2018.

[18] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[19] Q. D. Vo and P. De, "A survey of fingerprint-based outdoor localization," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 491–506, 2015.

[20] ETSI, "Service requirements for the 5G system," European Telecommunications Standards Institute (ETSI), Technical specification (TS) 122.261, 04 2021, version 16.14.0.

[21] F. Liu, C. Masouros, A. P. Petropulu, H. Griffiths, and L. Hanzo, "Joint radar and communication design: Applications, state-of-the-art, and the road ahead," *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3834–3862, 2020.

[22] D. K. P. Tan, J. He, Y. Li, A. Bayesteh, Y. Chen, P. Zhu, and W. Tong, "Integrated sensing and communication in 6G: Motivations, use cases, requirements, challenges and future directions," in *2021 1st IEEE International Online Symposium on Joint Communications & Sensing (JC&S)*. IEEE, 2021, pp. 1–6.

[23] F. Liu, Y. Cui, C. Masouros, J. Xu, T. X. Han, Y. C. Eldar, and S. Buzzi, "Integrated sensing and communications: Toward dual-functional wireless networks for 6g and beyond," *IEEE journal on selected areas in communications*, vol. 40, no. 6, pp. 1728–1767, 2022.

[24] F. Wen, H. Wymeersch, B. Peng, W. P. Tay, H. C. So, and D. Yang, "A survey on 5G massive MIMO localization," *Digital Signal Processing*, vol. 94, pp. 21–28, 2019.

[25] H. Wymeersch and G. Seco-Granados, "Radio localization and sensing—part ii: State-of-the-art and challenges," *IEEE Communications Letters*, vol. 26, no. 12, pp. 2821–2825, 2022.

[26] M. P. Wylie and J. Holtzman, "The non-line of sight problem in mobile location estimation," in *Proceedings of ICUPC-5th International Conference on Universal Personal Communications*, vol. 2. IEEE, 1996, pp. 827–831.

[27] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network-based wireless location: challenges faced in developing techniques for accurate wireless location information," *IEEE signal processing magazine*, vol. 22, no. 4, pp. 24–40, 2005.

[28] K. Witrisal, P. Meissner, E. Leitinger, Y. Shen, C. Gustafson, F. Tufvesson, K. Haneda, D. Dardari, A. F. Molisch, A. Conti, and M. Z. Win, "High-accuracy localization for assisted living: 5G systems will turn multipath channels from foe to friend," *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 59–70, 2016.

[29] D. Burghal, A. T. Ravi, V. Rao, A. A. Alghafis, and A. F. Molisch, "A comprehensive survey of machine learning based localization with wireless signals," *CoRR*, vol. abs/2012.11171, 2020. [Online]. Available: https://arxiv.org/abs/2012.11171

[30] X. Wang, L. Gao, S. Mao, and S. Pandey, "DeepFi: Deep learning for indoor fingerprinting using channel state information," in *2015 IEEE wireless communications and networking conference (WCNC)*. IEEE, 2015, pp. 1666–1671.

[31] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based positioning," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2017, pp. 1–6.

[32] A. Niitsoo, T. Edelhäußer, and C. Mutschler, "Convolutional neural networks for position estimation in TDOA-based locating systems," in *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. IEEE, 2018, pp. 1–8.

[33] J. Gante, G. Falcao, and L. Sousa, "Deep learning architectures for accurate millimeter wave positioning in 5G," *Neural Processing Letters*, vol. 51, no. 1, pp. 487–514, 2020.

[34] S. De Bast, A. P. Guevara, and S. Pollin, "CSI-based positioning in massive MIMO systems using convolutional neural networks," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. IEEE, 2020, pp. 1–5.

[35] R. Ayyalasomayajula, A. Arun, C. Wu, S. Sharma, A. R. Sethi, D. Vasisht, and D. Bharadia, "Deep learning based wireless localization for indoor navigation," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 1–14.

[36] R. Zekavat and R. M. Buehrer, *Handbook of position location: Theory, practice and advances*. John Wiley & Sons, 2011, vol. 27.

[37] M. Rupp and S. Schwarz, "An LS localisation method for massive MIMO transmission systems," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 4375–4379.

[38] M. Arnold, J. Hoydis, and S. ten Brink, "Novel massive MIMO channel sounding data applied to deep learning-based indoor positioning," in *SCC 2019; 12th International ITG Conference on Systems, Communications and Coding*. VDE, 2019.

[39] M. T. Hoang, B. Yuen, K. Ren, X. Dong, T. Lu, R. Westendorp, and K. Reddy, "A CNN-LSTM quantifier for single access point CSI indoor localization," *arXiv preprint arXiv:2005.06394*, 2020.

[40] A. Adhikary, J. Nam, J.-Y. Ahn, and G. Caire, "Joint spatial division and multiplexing—the large-scale array regime," *IEEE transactions on information theory*, vol. 59, no. 10, pp. 6441–6463, 2013.

[41] H. Q. Ngo, E. G. Larsson, and T. L. Marzetta, "Energy and spectral efficiency of very large multiuser MIMO systems," *IEEE Transactions on Communications*, vol. 61, no. 4, pp. 1436–1449, 2013.

[42] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An overview of massive MIMO: Benefits and challenges," *IEEE journal of selected topics in signal processing*, vol. 8, no. 5, pp. 742–758, 2014.

[43] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Transactions on Wireless Communications*, vol. 9, no. 11, pp. 3590–3600, 2010.

[44] S. Schwarz and S. Pratschner, "Multiple antenna systems in mobile 6G: Directional channels and robust signal processing," *IEEE Communications Magazine*, vol. 61, no. 4, pp. 64–70, 2023.

[45] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, pp. 276–280, 1986.

[46] H. Krim and M. Viberg, "Two decades of array signal processing research: the parametric approach," *IEEE signal processing magazine*, vol. 13, no. 4, pp. 67–94, 1996.

[47] W. Liu, M. Haardt, M. S. Greco, C. F. Mecklenbräuker, and P. Willett, "Twenty-five years of sensor array and multichannel signal processing: A review of progress to date and potential research directions," *IEEE Signal Processing Magazine*, vol. 40, no. 4, pp. 80–91, 2023.

[48] N. Garcia, H. Wymeersch, E. G. Larsson, A. M. Haimovich, and M. Coulon, "Direct localization for massive MIMO," *IEEE Transactions on Signal Processing*, vol. 65, no. 10, pp. 2475–2487, 2017.

[49] X. Sun, C. Wu, X. Gao, and G. Y. Li, "Fingerprint-based localization for massive MIMO-OFDM system with deep convolutional neural networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 10 846–10 857, 2019.

[50] S. De Bast and S. Pollin, "MaMIMO CSI-based positioning using CNNs: Peeking inside the black box," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2020, pp. 1–6.

[51] X.-H. You, D.-M. Wang, B. Sheng, X.-Q. Gao, X.-S. Zhao, and M. Chen, "Cooperative distributed antenna systems for mobile communications [coordinated and distributed MIMO]," *IEEE Wireless Communications*, vol. 17, no. 3, pp. 35–43, 2010.

[52] S. Schwarz, "Remote radio head assignment and beamforming in dynamic distributed antenna systems," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[53] V. Savic and E. G. Larsson, "Fingerprinting-based positioning in distributed massive MIMO systems," in *2015 IEEE 82nd vehicular technology conference (VTC2015-Fall)*. IEEE, 2015, pp. 1–5.

[54] K. S. V. Prasad, E. Hossain, and V. K. Bhargava, "Machine learning methods for RSS-based user positioning in distributed massive MIMO," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 8402–8417, 2018.

[55] X. Wang, X. Wang, and S. Mao, "Deep convolutional neural networks for indoor localization with CSI images," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 316–327, 2020.

[56] X. Wang, M. Patil, C. Yang, S. Mao, and P. A. Patel, "Deep convolutional Gaussian Processes for Mmwave outdoor localization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 8323–8327.

[57] K. S. V. Prasad, E. Hossain, V. K. Bhargava, and S. Mallick, "Analytical approximation-based machine learning methods for user positioning in distributed massive mimo," *IEEE Access*, vol. 6, pp. 18 431–18 452, 2018.

[58] J. Qiu, K. Xu, and Z. Shen, "Cooperative fingerprint positioning for cell-free massive MIMO systems," in *2020 International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2020, pp. 382–387.

[59] C. Wei, K. Xu, Z. Shen, X. Xia, W. Xie, L. Chen, and J. Xu, "Joint AOA-RSS fingerprint based localization for cell-free massive MIMO systems," in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*. IEEE, 2020, pp. 590–595.

[60] Y. Bengio, J.-f. Paiement, P. Vincent, O. Delalleau, N. Roux, and M. Ouimet, "Out-of-sample extensions for LLE, ISOMAP, MDS, Eigenmaps, and spectral clustering," *Advances in neural information processing systems*, vol. 16, 2003.

[61] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[62] N. Patwari, A. O. Hero, M. Perkins, N. S. Correal, and R. J. O'dea, "Relative location estimation in wireless sensor networks," *IEEE Transactions on signal processing*, vol. 51, no. 8, pp. 2137–2148, 2003.

[63] G. Morral and P. Bianchi, "Distributed on-line multidimensional scaling for self-localization in wireless sensor networks," *Signal processing*, vol. 120, pp. 88–98, 2016.

[64] Q. Song, S. Guo, X. Liu, and Y. Yang, "CSI amplitude fingerprinting-based NB-IoT indoor localization," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1494–1504, 2017.

[65] C. Studer, S. Medjkouh, E. Gönültaş, T. Goldstein, and O. Tirkkonen, "Channel charting: Locating users within the radio environment using channel state information," *IEEE Access*, vol. 6, pp. 47 682–47 698, 2018.

[66] J. Deng, W. Shi, J. Hu, and X. Jiao, "Semi-supervised t-SNE for millimeter-wave wireless localization," in *2021 7th International Conference on Computer and Communications (ICCC)*. IEEE, 2021, pp. 1015–1019.

[67] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European conference on computer vision*. Springer, 2016, pp. 850–865.

[68] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[69] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[70] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "Siamese" time delay neural network," in *Advances in neural information processing systems*, 1994, pp. 737–744.

[71] P. Ferrand, A. Decurninge, L. G. Ordoñez, and M. Guillaud, "Triplet-based wireless channel charting: Architecture and experiments," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2361–2373, 2021.

[72] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten ZIP code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[73] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[74] ETSI, "Study on new radio (NR) access technology," European Telecommunications Standards Institute (ETSI), Technical specification (TS) 38.912, 04 2021, version 15.0.0.

[75] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[76] P. Ferrand, A. Decurninge, and M. Guillaud, "DNN-based localization from channel estimates: Feature design and experimental results," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[77] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[78] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[79] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[80] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 076–10 085.

[81] B. Zhang, H. Sifaou, and G. Y. Li, "CSI-fingerprinting indoor localization via attention-augmented residual convolutional neural network," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[82] Y. Ruan, L. Chen, X. Zhou, G. Guo, and R. Chen, "Hi-Loc: Hybrid indoor localization via enhanced 5G NR CSI," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–15, 2022.

[83] A. Salihu, S. Schwarz, and M. Rupp, "Towards scalable uncertainty aware DNN-based wireless localisation," in *2021 29th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 1706–1710.

[84] ——, "Learning-based remote radio head selection and localization in distributed antenna system," in *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. IEEE, 2022, pp. 65–70.

[85] ——, "Semi-supervised localisation utilizing CSI at large antenna array base stations," in *WSA 2020; 24th International ITG Workshop on Smart Antennas*, 2020, pp. 1–5.

[86] A. Salihu, S. Schwarz, A. Pikrakis, and M. Rupp, "Low-dimensional representation learning for wireless CSI-based localisation," in *16th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)(50308)*. IEEE, 2020, pp. 1–6.

[87] A. Salihu, S. Schwarz, and M. Rupp, "Attention aided CSI wireless localization," in *2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, 2022, pp. 1–5.

[88] A. Salihu, M. Rupp, and S. Schwarz, "Self-supervised and invariant representations for wireless localization," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2024.

[89] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free massive mimo versus small cells," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1834–1850, 2017.

[90] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE communications magazine*, vol. 52, no. 2, pp. 186–195, 2014.

[91] J. Jose, A. Ashikhmin, T. L. Marzetta, and S. Vishwanath, "Pilot contamination and precoding in multi-cell TDD systems," *IEEE Transactions on Wireless Communications*, vol. 10, no. 8, pp. 2640–2651, 2011.

[92] P. Almers, E. Bonek, A. Burr, N. Czink, M. Debbah, V. Degli-Esposti, H. Hofstetter, P. Kyösti, D. Laurenson, G. Matz *et al.*, "Survey of channel and radio propagation models for wireless mimo systems," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, pp. 1–19, 2007.

[93] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave MIMO systems," *IEEE journal of selected topics in signal processing*, vol. 10, no. 3, pp. 436–453, 2016.

[94] S. Haykin and K. J. R. Liu, *Wireless Communication and Sensing in Multipath Environments Using Multiantenna Transceivers*, 2009, pp. 115–170.

[95] H. Wymeersch and G. Seco-Granados, "Radio localization and sensing–part I: Fundamentals," *IEEE Communications Letters*, 2022.

[96] P. Series, "Effects of building materials and structures on radiowave propagation above about 100 MHz," *Recommendation ITU-R*, pp. 2040–1, 2015.

[97] I. Ree, "P. 838, specific attenuation model for rain for use in prediction methods," *International Telecommunication Union, Geneva*, 1997.

[98] A. K. Jain, J. Mao, and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.

[99] D. A. Roberts, S. Yaida, and B. Hanin, *The principles of deep learning theory*. Cambridge University Press Cambridge, MA, USA, 2022.

[100] D. J. MacKay, "Probable networks and plausible predictions-a review of practical bayesian methods for supervised neural networks," *Network: computation in neural systems*, vol. 6, no. 3, p. 469, 1995.

[101] J. M. Hernández-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," in *International conference on machine learning*. PMLR, 2015, pp. 1861–1869.

[102] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[103] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, 2016, pp. 1050–1059.

[104] A. Loquercio, M. Segu, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3153–3160, 2020.

[105] C. M. Bishop, "Mixture density networks," Tech. Rep., 1994.

[106] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7144–7153.

[107] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.

[108] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[109] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[110] A. Alkhateeb, "DeepMIMO: A generic deep learning dataset for millimeter wave and massive MIMO applications," *arXiv preprint arXiv:1902.06435*, 2019.

[111] S. Pratschner, T. Blazek, E. Zöchmann, F. Ademaj, S. Caban, S. Schwarz, and M. Rupp, "A spatially consistent MIMO channel model with adjustable K factor," *IEEE Access*, vol. 7, pp. 110 174–110 186, 2019.

[112] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," *Advances in neural information processing systems*, vol. 28, pp. 2575–2583, 2015.

[113] M. F. Balın, A. Abid, and J. Zou, "Concrete autoencoders: Differentiable feature selection and reconstruction," in *International Conference on Machine Learning*. PMLR, 2019, pp. 444–453.

[114] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[115] E. J. Gumbel, "Statistical theory of extreme values and some practical applications," *NBS Applied Mathematics Series*, vol. 33, 1954.

[116] C. J. Maddison, D. Tarlow, and T. Minka, "A* sampling," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[117] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[118] C. P. Robert, G. Casella, and G. Casella, *Monte Carlo statistical methods*. Springer, 1999, vol. 2.

[119] D. G. Lewis and W. A. Gale, "W. a sequential algorithm for training text classifiers," in *Proceedings of SIGIR-94*, 1994, pp. 3–12.

[120] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *International conference on machine learning*. PMLR, 2017, pp. 1183–1192.

[121] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.

[122] L. Van Der Maaten, E. Postma, and J. Van den Herik, "Dimensionality reduction: a comparative review," *J Mach Learn Res*, vol. 10, no. 66-71, p. 13, 2009.

[123] C. K. Williams, "On a connection between kernel PCA and metric multidimensional scaling," *Machine Learning*, vol. 46, no. 1-3, pp. 11–19, 2002.

[124] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Trans. Comput.*, vol. 18, no. 5, pp. 401–409, May 1969. [Online]. Available: https://doi.org/10.1109/T-C.1969.222678

[125] D. Moltchanov, "Distance distributions in random networks," *Ad Hoc Networks*, vol. 10, no. 6, pp. 1146–1166, 2012.

[126] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[127] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[128] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A decomposable attention model for natural language inference," *arXiv preprint arXiv:1606.01933*, 2016.

[129] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[130] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 4171–4186.

[131] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[132] K. Guan, H. Yi, D. He, B. Ai, and Z. Zhong, "Towards 6G: Paradigm of realistic terahertz channel modeling," *China Communications*, vol. 18, no. 5, pp. 1–18, 2021.

[133] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[134] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

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

[136] MATLAB, *version 9.11.0 (R2021b)*. Natick, Massachusetts: The MathWorks Inc., 2021.

[137] K. Guan, Z. Zhong, B. Ai, and T. Kurner, "Deterministic propagation modeling for the realistic high-speed railway environment," in *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*. IEEE, 2013, pp. 1–5.

[138] Q. Zhao and J. Li, "Rain attenuation in millimeter wave ranges," *2006 7th International Symposium on Antennas, Propagation & EM Theory*, pp. 1–4, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:18767832

[139] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "WAV2VEC 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[140] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar *et al.*, "Bootstrap your own latent-a new approach to self-supervised learning," *Advances in neural information processing systems*, vol. 33, pp. 21 271–21 284, 2020.

[141] A. Bardes, J. Ponce, and Y. LeCun, "VICREG: Variance-invariance-covariance regularization for self-supervised learning," *arXiv preprint arXiv:2105.04906*, 2021.

[142] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9630–9640.

[143] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.

[144] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of machine learning research*, vol. 9, 2008.

[145] W.-L. Chin, C.-C. Hsieh, D. Shiung, and T. Jiang, "Intelligent indoor positioning based on artificial neural networks," *IEEE Network*, vol. 34, no. 6, pp. 164–170, 2020.

[146] W. Liu, M. Jia, Z. Deng, and C. Qin, "MHSA-EC: An indoor localization algorithm fusing the multi-head self-attention mechanism and effective CSI," *Entropy*, vol. 24, no. 5, p. 599, 2022.

[147] R. Hadani, S. Rakib, M. Tsatsanis, A. Monk, A. J. Goldsmith, A. F. Molisch, and R. Calderbank, "Orthogonal time frequency space modulation," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–6.

[148] D. Newman, S. Hettich, C. Blake, and C. Merz, "UCI repository of machine learning databases," 1998. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[149] C. Mehlführer, J. Colom Ikuno, M. Šimko, S. Schwarz, M. Wrulich, and M. Rupp, "The Vienna LTE simulators-enabling reproducibility in wireless communications research," *EURASIP Journal on Advances in Signal Processing*, vol. 2011, pp. 1–14, 2011.

[150] M. Rupp, S. Schwarz, M. Taranetz *et al.*, "The Vienna LTE-advanced simulators," *https://link. springer. com/book/10.1007% 2F978-981-10-0617-3*, 2016.

[151] R. Balestriero, M. Ibrahim, V. Sobal, A. Morcos, S. Shekhar, T. Goldstein, F. Bordes, A. Bardes, G. Mialon, Y. Tian *et al.*, "A cookbook of self-supervised learning," *arXiv preprint arXiv:2304.12210*, 2023.

[152] J. Von Kügelgen, Y. Sharma, L. Gresele, W. Brendel, B. Schölkopf, M. Besserve, and F. Locatello, "Self-supervised learning with data augmentations provably

isolates content from style," *Advances in neural information processing systems*, vol. 34, pp. 16 451–16 467, 2021.

[153] R. Balestriero and Y. LeCun, "Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods," *Advances in Neural Information Processing Systems*, vol. 35, pp. 26 671–26 685, 2022.