THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

# VECTOR EMBEDDINGS TO ESTIMATE ANGLE OF ARRIVAL FOR WIRELESS SIGNALS

BHARAVI MISRA
SPRING 2025

A thesis
submitted in partial fulfillment
of the requirements
for a baccalaureate degree
in Computer Engineering
with honors in Computer Engineering

Reviewed and approved* by the following:

Jinhgui Chen
Assistant Professor of Information Sciences and Technology
Thesis Supervisor

John (Jack) Sampson
Associate Department Head of Computer Science and Engineering
Honors Adviser

*Signatures are on file in the Schreyer Honors College.

# Abstract

This thesis explores the application of deep learning techniques to the problem of angle of arrival (AoA) estimation in wireless communication systems. I propose a novel neural architecture that leverages `Data2VecAudio`, a self-supervised model pre-trained on speech, vision, and text, as a feature extractor for complex antenna array signals. The model is trained end-to-end using synthetic data generated in MATLAB and is evaluated against classical signal processing baselines, including MUSIC and Beamscan. Experimental results demonstrate that the proposed model performs comparably to traditional methods and exhibits strong generalization capabilities, despite being trained on a limited and simplified dataset. Notably, the model is able to infer meaningful spatial patterns from raw complex I/Q data, even with variable-length inputs. However, several limitations remain, including reliance on simulated data, simplified noise models, fixed source counts, and a computationally intensive architecture. These findings highlight both the potential and the current challenges of applying general-purpose vector embeddings to wireless signal processing tasks, and provide a foundation for future research into more efficient and scalable architectures suitable for deployment in real-world environments.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgements

I would like to thank my advisors Professor Jinghui Chen and Professor Jack Sampson for their help navigating the thesis process, as well as Professor Mahanth Gowda for introducing me to the interesting intersection between wireless communications and deep learning. I would also like to thank the Schreyer Honors College and the Pennsylvania State University for its support in my academic and professional endeavors throughout my unergraduate experience.

# Chapter 1

# Introduction

## 1.1 Motivation

### 1.1.1 Problem Statement

Wireless communication plays a critical role in modern networking, enabling a wide range of applications from cellular communications to IoT networks and autonomous systems [11]. One of the fundamental challenges in wireless systems is the estimation of the Angle of Arrival (AoA) of received signals, which is essential for localization, beamforming, and interference mitigation. Traditional AoA estimation techniques, such as MUSIC (Multiple Signal Classification), ESPRIT (Estimation of Signal Parameters via Rotational Invariance Techniques), and subspace-based methods, often rely on precise array calibration, high computational costs, and strong assumptions about the signal environment. These constraints limit their applicability in dynamic, real-world scenarios where multipath interference, noise, and hardware imperfections introduce significant variations [11].

In recent years, deep learning and machine learning approaches have emerged as powerful alternatives to classical signal processing methods [13]. Among these, vector embedding models—such as Data2Vec and other self-supervised learning frameworks—offer a promising direction for capturing rich signal representations without requiring explicit feature engineering [2]. These models learn generalizable embeddings of input signals by leveraging large-scale data, making them robust against variations in signal distortions and environmental conditions. By encoding wireless signals into structured latent spaces, vector embedding models can facilitate efficient and scalable AoA estimation, potentially outperforming traditional methods in complex propagation environments.

This thesis explores the application of vector embedding models for AoA estimation of wireless signals. Specifically, it investigates how self-supervised learning techniques can be adapted to extract meaningful features from raw RF signals captured by elements in an antenna array for accurate angle estimation across diverse scenarios. The study aims to address key research questions, including:

1. How effectively can embedding-based models generalize across different antenna configurations and environments?

2. Can pre-trained embeddings models trained for human speech recognition generalize well to classical signal processing tasks?

3. How do embedding-based AoA estimation techniques compare to classical baselines in terms of accuracy and robustness across a wide range of signal conditions?

The remainder of this thesis is structured as follows: The rest of Chapter 1 introduces the motivation behind this research, defining the problem and reviewing existing technologies, including classical AoA estimation techniques and deep learning-based approaches. It also provides background on beamforming, the MUSIC algorithm, and embedding models. Chapter 2 details the methodology, covering data collection, model architecture, training strategy, and evaluation metrics used to assess performance. Chapter 3 focuses on the implementation, discussing software frameworks, system design, and the model training pipeline. Chapter 4 presents the results, analyzing performance, generalization, and robustness while identifying limitations and directions for future research.

By bridging the gap between machine learning-based representations and wireless signal processing, this research aims to contribute to the development of more robust and scalable AoA estimation techniques, paving the way for enhanced localization and beamforming capabilities in next-generation wireless systems.

### 1.1.2 Existing Technology

Traditional Angle of Arrival (AoA) estimation methods have played a crucial role in wireless communication and signal processing, with techniques such as beamforming, MUSIC (Multiple Signal Classification), and ESPRIT (Estimation of Signal Parameters via Rotational Invariance Techniques) being widely used. Beamforming is a spatial filtering technique that enhances signals arriving from a specific direction while suppressing interference, typically implemented using phased antenna arrays [16]. However, its resolution is constrained by the array aperture and the number of antennas. The MUSIC algorithm, a subspace-based method, exploits the eigenstructure of the covariance matrix to estimate AoA by separating the signal and noise subspaces, achieving high resolution under favorable conditions [15]. Nevertheless, MUSIC requires precise array calibration and a sufficient signal-to-noise ratio (SNR) to perform effectively. ESPRIT, another high-resolution method, leverages the rotational invariance property of the signal subspace to estimate AoA while reducing computational complexity compared to MUSIC [14]. Despite their effectiveness, these classical approaches struggle in non-ideal environments with multipath propagation, correlated signals, and hardware imperfections, motivating the exploration of data-driven techniques such as deep learning and embedding models for robust AoA estimation.

Recent advancements in AoA estimation have built upon classical algorithms by incorporating additional signal processing techniques and leveraging Channel State Information (CSI) for improved accuracy [11]. Methods such as ArrayTrack and SpotFi refine traditional approaches by mitigating multipath interference and enhancing robustness in real-world environments. ArrayTrack improves AoA estimation by utilizing fine-grained CSI data along with a novel tracking algorithm to localize wireless transmitters more accurately, even in dynamic indoor environments [18]. It refines classical beamforming techniques by leveraging signal strength and phase differences across multiple antennas to enhance resolution. SpotFi, on the other hand, extends the capabilities of the MUSIC algorithm by employing CSI-based subcarrier-level information to separate multipath components, thereby achieving sub-meter localization accuracy without requiring specialized hardware [6]. By leveraging frequency diversity and phase sanitization techniques, SpotFi significantly enhances robustness against environmental distortions. These advancements demonstrate the potential of CSI-based signal processing to overcome the limitations of classical AoA estimation methods, paving the way for further innovations using machine learning and self-supervised learning models.

Deep learning has recently emerged as a powerful tool for AoA estimation, offering both end-to-end solutions and hybrid approaches that augment traditional signal processing techniques [9]. End-to-end deep learning models bypass explicit feature extraction by directly learning mappings from raw wireless signal inputs, such as Channel State Information (CSI) or received signal strength (RSS), to AoA estimates. These models, typically implemented using convolutional neural networks (CNNs) or recurrent neural networks (RNNs), can automatically learn spatial and temporal correlations in the signal, improving robustness to noise, multipath effects, and hardware imperfections [5], [8], [12], [17]. In contrast, hybrid approaches integrate deep learning into

specific stages of classical pipelines, enhancing performance without entirely replacing traditional algorithms. For example, deep neural networks (DNNs) have been used to refine covariance matrix estimation in MUSIC, optimize denoising and phase correction in CSI-based systems, or improve subspace decomposition techniques in ESPRIT [3], [4], [9]. Finally, the most recent works, particularly in sound source localization, have leveraged transformers and embedding models to capture complex spatial relationships and temporal dependencies for accurate Angle of Arrival (AoA) estimation [5], [7]. By leveraging both data-driven learning and domain knowledge from classical signal processing, these methods achieve superior accuracy and generalization across diverse deployment environments. The growing adoption of self-supervised learning techniques, such as embedding models, further enhances the ability of deep learning-based AoA estimation to adapt to unseen signal variations without extensive labeled data, marking a significant evolution in wireless localization technology.

## 1.2 Background

### 1.2.1 Angle of Arrival (AoA) Estimation Overview

Angle of Arrival (AoA) estimation is a fundamental problem in wireless communication and signal processing, concerned with determining the direction from which a received signal originates relative to a reference point, typically an antenna array. AoA estimation plays a crucial role in applications such as source localization, beamforming, and interference mitigation in wireless networks, radar, and acoustic sensing.

The core principle behind AoA estimation relies on the time delay or phase difference of a signal arriving at multiple spatially separated antennas. When a plane wave propagates towards an array of sensors, the signal reaches each element at slightly different times, creating measurable differences in phase or time-of-arrival. By exploiting these differences, it is possible to infer the direction of the incoming signal. The resolution and accuracy of AoA estimation depend on factors such as the number of antenna elements, their spacing, the signal's wavelength, and environmental conditions such as multipath propagation and noise.

A fundamental model for AoA estimation assumes a uniform linear array (ULA) of antennas, where an incoming signal with wavelength $\lambda$ impinges on the array at an angle $\theta$ relative to the array axis. Figure 1.1 illustrates an incoming signal reaching a ULA.

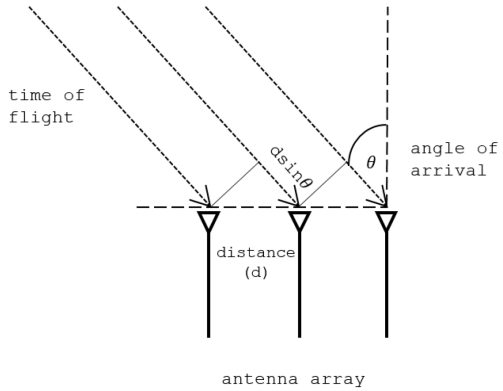


Figure 1.1: Signal impinging on a linear antenna array [1].

The time delay between consecutive antenna elements separated by a distance $d$ is given by:

$$\tau = \frac{d \sin \theta}{c} \tag{1.1}$$

where $c$ is the speed of propagation (e.g., the speed of light for RF signals or the speed of sound for acoustic signals). This delay translates into a phase shift, which forms the basis for estimating $\theta$. By analyzing the received signals across the antenna array, mathematical techniques such as beamforming, subspace decomposition (e.g., MUSIC and ESPRIT), or deep learning-based models can be used to infer the AoA with high accuracy.

AoA estimation is an inherently ill-posed problem when a limited number of antennas are used, particularly in the presence of multipath interference where multiple copies of the same signal arrive from different directions. To overcome these challenges, modern techniques employ high-resolution algorithms, spatial smoothing, and data-driven methods to enhance estimation accuracy.

### 1.2.2 Beamforming

Beamforming is a signal processing technique used to direct and enhance the reception or transmission of signals in specific directions by leveraging multiple antennas or sensor elements [16]. At its core, beamforming exploits the constructive and destructive interference of waves received or transmitted at an array of antennas, allowing the system to focus on signals arriving from or departing toward a particular direction.

The close relationship between beamforming and AoA estimation arises from their shared reliance on spatial signal processing. In AoA estimation, the goal is to determine the direction from which an incoming signal originates. In contrast, beamforming uses this directional information to enhance signals from desired directions while suppressing interference from others. Both techniques rely on the same fundamental principle: the phase difference observed at different antenna elements due to the varying propagation paths of the incoming wavefront.

A traditional beamforming model considers a ULA with $N$ antennas spaced at a distance $d$. If a signal impinges on the array from an angle $\theta$, the received signals at different antennas exhibit a progressive phase shift. By applying phase shifts and weights to the received signals before summing them, a beamformer can reinforce signals arriving from the desired angle while attenuating others. The output of a beamformer is given by:

$$y(t) = \sum_{n=1}^{N} w_n x_n(t), \tag{1.2}$$

where $x_n(t)$ represents the signal received at the $n$-th antenna and $w_n$ is the complex weight applied to steer the beam. By adjusting these weights dynamically, the array can "scan" different angles, forming spatially selective beams.

Beamforming enhances AoA estimation by allowing a system to focus on specific directions and improve the signal-to-noise ratio (SNR) of received signals. Classical beamforming approaches, such as Delay-and-Sum and Minimum Variance Distortionless Response (MVDR), optimize the weight vector to maximize reception from a specific angle. More advanced techniques, such as adaptive beamforming, incorporate real-time environmental feedback to optimize the beam pattern dynamically.

The classic beamformer estimates the AoA by scanning different spatial directions and measuring the power of the received signal. Given a ULA with $N$ antennas spaced at a distance $d$, the received signal at the $n$-th antenna for an incoming plane wave from direction $\theta$ is phase-shifted due to the propagation delay. The beamformer applies a weight vector $\mathbf{w} = [w_1, w_2, ..., w_N]$ to steer the array response in a particular direction, forming the beamformer output as:

$$y(\theta) = \sum_{n=1}^{N} w_n x_n e^{-j\frac{2\pi}{\lambda} d_n \sin\theta}, \tag{1.3}$$

where $x_n$ is the received signal at the $n$-th antenna, $\lambda$ is the signal wavelength, and $d_n$ represents the antenna spacing. By scanning over a range of angles and computing the beamformer output power, the AoA can be estimated as the angle $\theta$ that maximizes the received signal power:

$$\hat{\theta} = \arg\max_{\theta} |y(\theta)|^2. \tag{1.4}$$

This method, known as Delay-and-Sum beamforming, provides a simple yet effective approach for AoA estimation but suffers from limited resolution, especially when signals arrive from closely spaced directions.

Figure 1.2 illustrates the spectrum generated by a beamforming technique called the classical beamscanner. As demonstrated, it can correctly identify AoA for 3 distant sources, but lacks resolution.

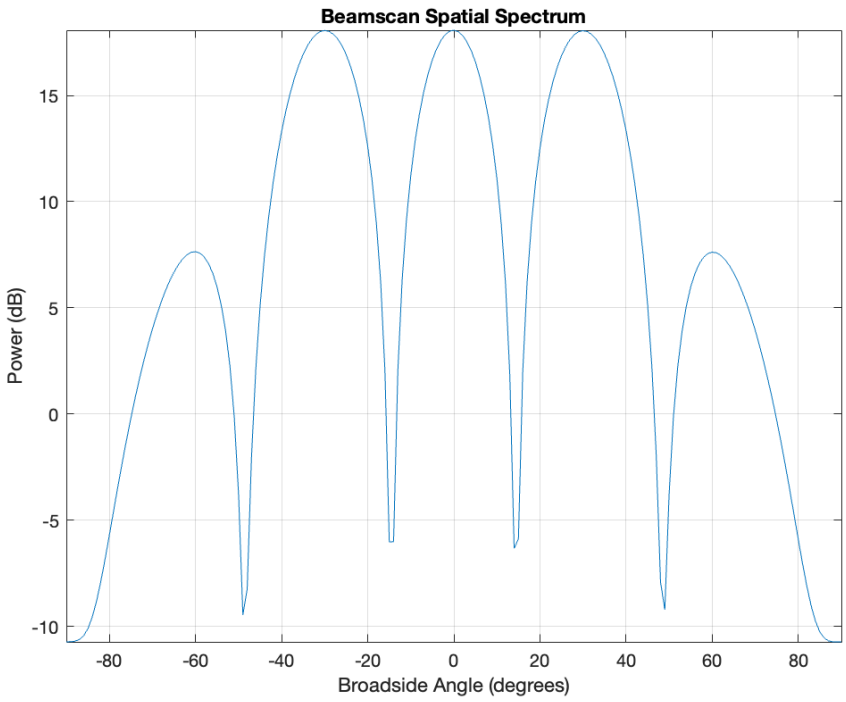


Figure 1.2: Beamscan spatial spectrum produced by an 8-element uniform linear array (ULA) operating at 5.25 GHz. The estimator correctly identifies three signal directions of arrival (DoAs) at $[-30°, 0°, 30°]$ azimuth, using 200 snapshots and a noise power of 0.01.

In summary, beamforming and AoA estimation are intrinsically linked, as both rely on spatial signal processing to extract or enhance directional information. While AoA estimation determines where a signal is coming from, beamforming leverages this information to improve reception and mitigate interference, making it a crucial technique for modern communication and sensing systems.

### 1.2.3 MUSIC Algorithm

The Multiple Signal Classification (MUSIC) algorithm is a high-resolution subspace-based method for AoA estimation [15]. It exploits the eigenstructure of the received signal covariance matrix to separate signal and noise subspaces, allowing for precise AoA estimation even in the presence of noise.

#### Signal Model

Consider a uniform linear array (ULA) of $N$ antennas receiving $M$ narrowband plane waves from unknown angles $\theta_1, \theta_2, ..., \theta_M$. The received signal vector at time $t$ is:

$$\mathbf{x}(t) = \mathbf{A}(\boldsymbol{\theta})\mathbf{s}(t) + \mathbf{n}(t), \tag{1.5}$$

where $\mathbf{x}(t) \in \mathbb{C}^{N\times 1}$ is the received signal vector, $\mathbf{A}(\boldsymbol{\theta}) = [\mathbf{a}(\theta_1), ..., \mathbf{a}(\theta_M)] \in \mathbb{C}^{N\times M}$ is the

array steering matrix, $\mathbf{s}(t) \in \mathbb{C}^{M\times 1}$ is the source signal vector, and $\mathbf{n}(t) \in \mathbb{C}^{N\times 1}$ is additive noise. The steering vector for an incoming signal from angle $\theta$ is given by:

$$\mathbf{a}(\theta) = \begin{bmatrix} 1 \\ e^{-j2\pi d \sin\theta/\lambda} \\ \vdots \\ e^{-j2\pi(N-1)d\sin\theta/\lambda} \end{bmatrix}. \tag{1.6}$$

**Signal and Noise Subspaces**

The sample covariance matrix of the received signal is:

$$\mathbf{R}_x = \mathbb{E}[\mathbf{x}(t)\mathbf{x}^H(t)] = \mathbf{A}(\boldsymbol{\theta})\mathbf{R}_s\mathbf{A}^H(\boldsymbol{\theta}) + \sigma_n^2\mathbf{I}, \tag{1.7}$$

where $\mathbf{R}_s$ is the source covariance matrix and $\sigma_n^2\mathbf{I}$ is the noise power. By performing eigenvalue decomposition (EVD) on $\mathbf{R}_x$, we obtain eigenvectors corresponding to signal and noise subspaces. The noise subspace $\mathbf{E}_n$ consists of the eigenvectors associated with the smallest eigenvalues, and it is orthogonal to the signal subspace spanned by the steering vectors.

**MUSIC Spectrum and Root-MUSIC**

The MUSIC pseudo-spectrum is computed as:

$$P_{\text{MUSIC}}(\theta) = \frac{1}{\mathbf{a}^H(\theta)\mathbf{E}_n\mathbf{E}_n^H\mathbf{a}(\theta)}. \tag{1.8}$$

The AoA estimates correspond to the peaks of $P_{\text{MUSIC}}(\theta)$. Instead of searching over a grid of angles, **Root-MUSIC** reformulates the problem into a polynomial root-finding approach by solving for the roots of a characteristic polynomial, significantly improving computational efficiency.

**Spatial Smoothing for Correlated Sources**

When sources are correlated (e.g., due to multipath propagation), the rank of $\mathbf{R}_x$ reduces, affecting the MUSIC algorithm's performance. **Spatial smoothing** mitigates this issue by partitioning the array into overlapping subarrays, computing their covariance matrices, and averaging them:

$$\mathbf{R}_{\text{smoothed}} = \frac{1}{L}\sum_{l=1}^{L}\mathbf{R}_x^{(l)}, \tag{1.9}$$

where $L$ is the number of subarrays. This restores the full rank of the covariance matrix, enabling MUSIC to resolve correlated sources effectively.

Figure 1.3 illustrates the spectrum generated by the MUSIC algorithms. As demonstrated, it can correctly identify AoA for 3 distant sources, with much higher resolution compared to beamforming.

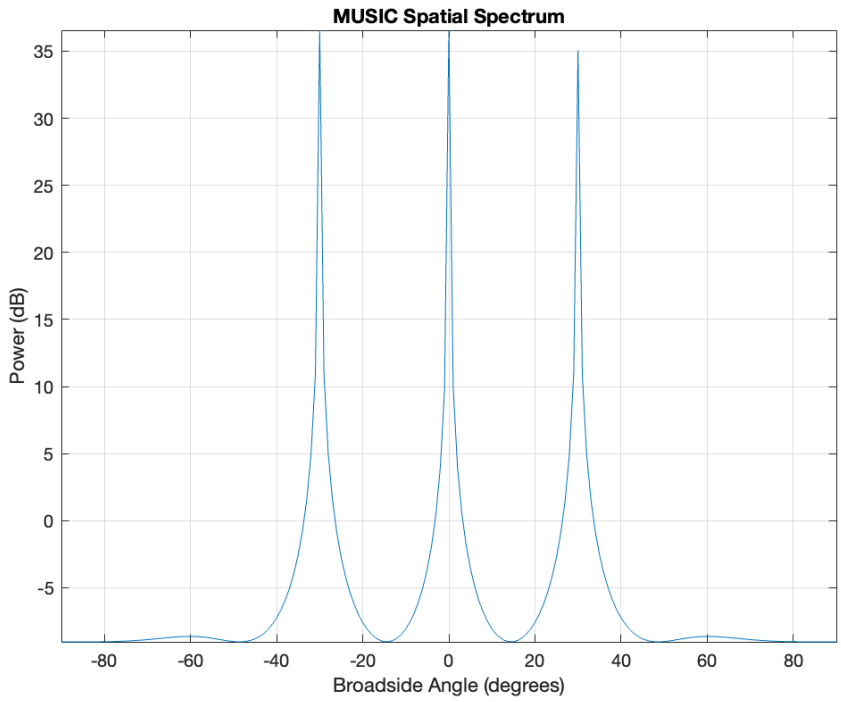


Figure 1.3: MUSIC spatial spectrum generated using an 8-element uniform linear array (ULA) at 5.25 GHz. The algorithm resolves three incoming signals with directions of arrival (DoAs) at $[-30°, 0°, 30°]$ azimuth azimuth, based on 200 snapshots. Compared to conventional beamforming, MUSIC provides higher resolution and sharper peaks for closely spaced sources.

MUSIC remains a widely used high-resolution AoA estimation method, providing superior resolution over classic beamforming but requiring accurate covariance estimation and sufficient SNR for reliable performance.

### 1.2.4 DNNs to Estimate Angle of Arrival

Deep neural networks (DNNs) have emerged as a powerful alternative to classical AoA estimation methods, leveraging data-driven learning to model complex signal propagation characteristics. Unlike traditional approaches that rely on explicit signal models and subspace decomposition, deep learning-based methods can extract features directly from raw received signals, enabling robust AoA estimation even in challenging environments with multipath effects and noise.

#### End-to-End Deep Learning Approaches

End-to-end deep learning approaches reduce the need for handcrafted feature extraction by directly learning the mapping from input data, such as raw Channel State Information (CSI), signal covariance matrices, or input samples, to AoA estimates. These methods typically employ convolutional neural networks (CNNs), recurrent neural networks (RNNs), or transformer-based architectures to capture spatial and temporal dependencies in the received signals [5], [11].

A common formulation involves treating the AoA estimation problem as a classification or regression task. Given an input signal representation $\mathbf{X}$, a neural network parameterized by $\theta$ learns a function $f_\theta(\mathbf{X})$ to predict the AoA $\hat{\theta}$:

$$\hat{\theta} = f_\theta(\mathbf{X}). \tag{1.10}$$

CNN-based models process received signals as structured images or spectrograms, leveraging spatial feature extraction to learn discriminative patterns associated with different AoAs. RNNs and Long Short-Term Memory (LSTM) networks capture sequential dependencies, particularly useful in dynamic environments where signal variations occur over time [17].

Figure 1.4 illustrates the model architecture for widely-cited deep CNN. It takes the signal array covariance matrix as the input and outputs a vector of discrete angle classifications.

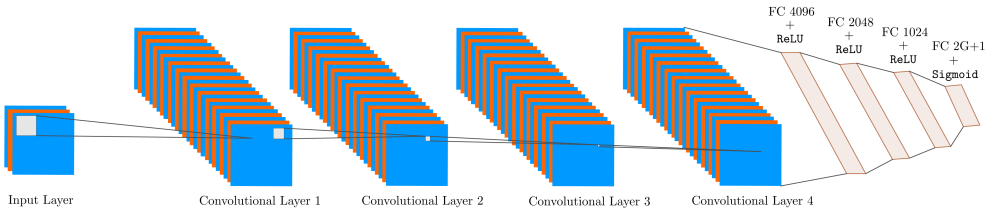


Figure 1.4: Model architecture for [12]. Takes signal array covariance matrix as input and outputs $n$-hot vector of discrete angle classifications.

### Advantages and Challenges

End-to-end deep learning approaches offer several advantages over traditional methods, including improved generalization in complex environments, reduced dependence on precise array calibration, and the ability to learn from large-scale datasets. However, they require extensive labeled training data, high computational resources, and careful model design to ensure robustness against unseen conditions. Additionally, interpretability remains a challenge, as deep learning models often function as black-box predictors without explicit insights into their decision-making process [5].

### Hybrid Deep Learning Approach for AoA Estimation

Another recently popular approach to the AoA estimation problem employs a hybrid deep learning approach that integrates DNNs into the classical subspace-based MUSIC algorithm. Traditional model-based techniques, such as MUSIC, rely on precise signal modeling and covariance matrix estimation, making them sensitive to model mismatches, correlated sources, and noise. To address these limitations, several authors propose augmenting MUSIC with neural networks to improve critical estimation steps while preserving the interpretability and structure of the original algorithm [3], [4], [9].

The approaches utilize deep learning to refine the estimation of the noise and signal subspaces, which are crucial for accurate AoA detection. A recurrent neural network (RNN) is typically employed to learn a pseudo-covariance matrix from raw measurements, replacing the empirical

covariance estimation in MUSIC. This enables the system to adaptively model complex signal conditions, mitigating the effects of correlated sources and low signal-to-noise ratios (SNRs). Additionally, a neural network-based peak finder replaces conventional spectral peak detection, allowing for improved resolution and differentiability, which facilitates end-to-end training.

Figure 1.5 illustrates the model architecture for [9]. It takes the signal array covariance matrix as the input and feeds the data through an RNN before doing Eigenvalue Decomposition (EVD) to get the spatial matrix used for the MUSIC spectrum. It then feeds the data through an intermediary fully connected neural network to estimate the number of sources, which is then used by the MUSIC algorithm to output possible directions of arrival.

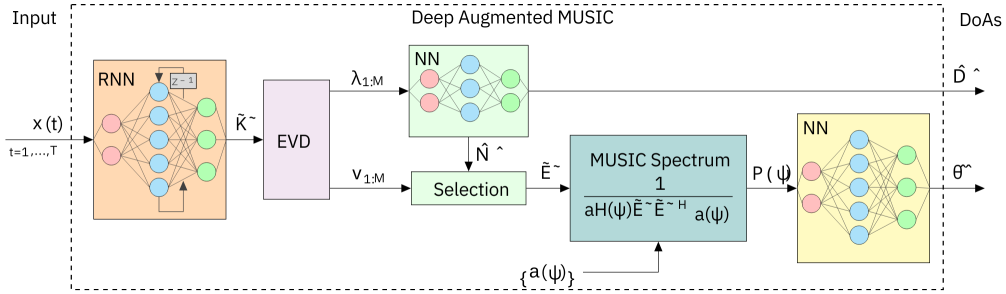


Figure 1.5: Model architecture for [9]. Takes signal array covariance matrix as input and outputs estimate of $\hat{D}$ sources and $\hat{\theta}$ angles.

The hybrid frameworks effectively combine domain knowledge from classical signal processing with the adaptability of deep learning. By integrating neural networks at key stages of the AoA estimation pipeline, these methods enhance resolution, generalization, and robustness to environmental uncertainties while maintaining the efficiency and interpretability of subspace-based approaches. Experimental results demonstrate that these deep-augmented MUSIC algorithm significantly outperforms traditional MUSIC in resolving coherent sources and broadband signals, offering a promising direction for future AoA estimation in dynamic and complex environments.

### 1.2.5 Embedding Models

Embedding models are machine learning techniques used to map high-dimensional data into lower-dimensional continuous vector spaces while preserving meaningful relationships. Originally developed for natural language processing (NLP), embeddings have since been applied to various domains, including computer vision, recommendation systems, and wireless signal processing. These models learn structured representations of data that capture semantic or spatial relationships, making them potentially useful for AoA estimation.

One of the foundational breakthroughs in embedding models is **Word2Vec**, which introduced two key architectures: *Skip-gram* and *Continuous Bag-of-Words (CBOW)* [10]. In the Skip-gram model, the objective is to predict the context words given a target word, optimizing the following loss function:

$$J = -\sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log P(w_{t+j}|w_t), \tag{1.11}$$

where $w_t$ represents a target word at position $t$, and $P(w_{t+j}|w_t)$ is the probability of predicting a context word within a window size $c$. Word2Vec learns vector representations by maximizing the likelihood of correct context predictions, resulting in embeddings where semantically similar words have similar vector representations.

A significant extension of embedding models is **Data2Vec**, a self-supervised learning approach that unifies embeddings across modalities such as text, audio, and vision [2]. Instead of relying on predefined token relationships, Data2Vec learns structured feature representations by predicting latent target embeddings from masked inputs, making it highly generalizable.

### Embedding Models for AoA Estimation

The principles of embedding learning can be adapted to AoA estimation by treating wireless signals as high-dimensional data and learning compact vector representations that preserve spatial and temporal features. Instead of manually engineering features, an embedding model can learn meaningful representations of received signals or covariance matrices, capturing essential patterns for robust AoA estimation. By leveraging self-supervised learning, embedding-based models can generalize across different antenna configurations and environmental conditions, making them promising for data-driven localization techniques [7].

Recent advancements suggest that embedding models, such as transformer-based architectures, can be integrated with deep learning frameworks to improve AoA estimation accuracy, particularly in multipath environments and low-SNR conditions. This approach bridges the gap between classical signal processing and modern machine learning, offering a scalable and interpretable alternative to traditional methods.

# Chapter 2

# Methodology

## 2.1 Datasets

### 2.1.1 Training Data

To train the AoA estimation model, I synthetically generate a dataset using a custom MATLAB function that simulates realistic wireless signal reception at a ULA. Each training example consists of a complex baseband signal matrix received at the array, along with the corresponding ground truth directions of arrival (DoAs) in radians.

The antenna array used in all simulations is a ULA comprising $M = 8$ isotropic elements with half-wavelength spacing. The system operates at a carrier frequency of $f_c = 5.25$ GHz, corresponding to a wavelength of $\lambda = c/f_c \approx 0.0571$ m, where $c$ is the speed of light. The array is aligned along a single spatial dimension, and element positions are normalized by $\lambda$ to simplify modeling in the far-field narrowband regime. This configuration is typical for wireless communication systems operating in the 5 GHz band (e.g., Wi-Fi) and enables spatial resolution suitable for direction-of-arrival estimation across a wide range of angles.

For each of the $n$ training samples, the following procedure is executed:

1. A random set of $K$ source angles $\boldsymbol{\theta} \in [-60^\circ, 60^\circ]^K$ is sampled uniformly to represent the true azimuth directions of incoming signals.

2. The number of temporal snapshots $T$ is selected from a predefined range $\{20, 30, \ldots, 200\}$, cycling through values to ensure variability across the dataset.

3. A signal-to-noise ratio (SNR) value is similarly chosen from the set $\{-20, -15, \ldots, 20\}$ dB.

4. For each source, narrowband far-field signals are synthesized using the function `sensorsig()`, which produces the received complex signal at the array based on the array geometry, snapshot count, DoAs, and noise power. Additive white Gaussian noise is injected to match the selected SNR.

5. The azimuth angles are converted to broadside angles using the function `az2broadside()`, and finally converted to radians to serve as ground truth AoA labels.

Each dataset sample contains:

- `X`: A complex matrix of size $M \times T$, where $M$ is the number of array elements and $T$ is the number of temporal snapshots.

- `DoAs`: A length-$K$ vector of broadside angles (in radians), corresponding to the ground truth AoAs.

Figure 2.1 illustrates the signal received by each of the 8 elements in the ULA. The image is taken from one of the data points in the training dataset.

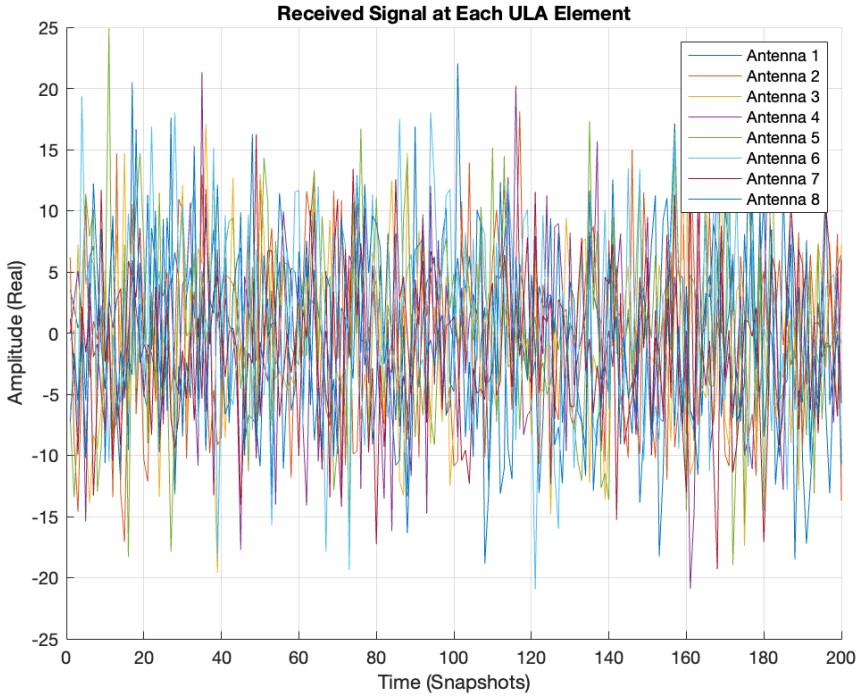


Figure 2.1: Signals received by each of the 8 antennas in ULA.

A fixed random seed is used for reproducibility (`rng(42)`). This dataset allows training and evaluation across a wide range of signal conditions, array snapshots, and AoA scenarios, promoting generalization in the learned model.

### 2.1.2 Test Data

To rigorously evaluate the generalization ability of the proposed AoA estimation model, I designed a set of diverse test datasets that each target specific signal conditions. These datasets are generated synthetically using MATLAB scripts that simulate complex baseband signals received at a ULA, consistent with the array geometry used in training. All datasets assume narrowband plane-wave propagation under the far-field assumption, with additive white Gaussian noise. A fixed random seed (`rng(42)`) ensures reproducibility across experiments.

#### Closely-Spaced Sources

The `generateCloselySpacedSources` script produces test samples containing multiple sources with small angular separation. This setup is particularly challenging due to the limited spatial resolution of the array. For each of the $n$ examples:

- A base azimuth angle is sampled uniformly from $[-60°, 60°]$.
- Angular separations $\delta$ between sources are selected from the range $[0, 0.5]$ radians, in increments of 0.05.

- Subsequent source angles are generated by incrementing the base azimuth by $\Delta$ (converted to degrees).
- The number of temporal snapshots is fixed at 200, and noise power is set to $0.01$.

Each sample contains the received signal matrix `X`, the true AoAs `DoAs` in radians, and the angular separation `Delta`.

**Variable Snapshot Count**

The `generateVariableSnapshotDataset` script evaluates the model's performance under varying observation durations. This is a realistic assessment of model performance because AoA measurements are most often taken from short cyclic redundancy check (CRC) segments in WiFi packet day. For each of the $n$ test examples:

- A fixed number of sources is placed at uniformly random azimuth angles within $[-60°, 60°]$.
- The number of snapshots $T$ is chosen from the set $\{20, 30, \ldots, 200\}$ to represent a wide range of temporal resolutions.
- The noise power is fixed at $0.01$.

Each sample includes the received signal `X`, the true AoAs `DoAs`, and the snapshot count `snapshots`.

**Variable Signal-to-Noise Ratio**

The `generateVariableSNRSignals` script generates test examples under a wide range of signal-to-noise ratio (SNR) conditions to assess the model's robustness to noise. For each example:

- The azimuth angles are sampled uniformly within $[-60°, 60°]$.
- The SNR is selected from the set $\{-20, -15, \ldots, 20\}$ dB.
- Noise power is computed based on a fixed signal power of 1 and the selected SNR.
- The number of snapshots is fixed at 200.

Each output includes the received signal matrix `X`, the true AoAs `DoAs`, and the associated SNR value `SNR`.

**Summary**

These test datasets introduce challenging and diverse signal scenarios—such as low SNR, few snapshots, and closely-spaced sources—that are not explicitly present in the training set. This design enables us to measure the generalization performance of the deep learning model across practical conditions encountered in real-world AoA estimation tasks.

## 2.2 Evaluated Models

### 2.2.1 Beamscan

To benchmark the performance of the proposed deep learning model, I include the classical Beamscan algorithm as a baseline for AoA estimation. The estimator is configured with the same 8-element ULA and a carrier frequency of 5.25 GHz. A fixed scan range from $-90°$ to $90°$ is used to compute the output spatial spectrum, from which the directions of arrival are estimated. The number of sources is specified in advance to extract the corresponding peaks.

The input to the Beamscan estimator is the complex-valued signal matrix $\mathtt{X}$ received across the array. The output is a set of estimated AoAs in degrees, which are subsequently converted to radians for comparison against ground truth. As a power-scanning method, Beamscan is relatively simple and less computationally demanding than subspace methods like MUSIC, making it a suitable lower-bound baseline for evaluating the benefits of deep learning approaches.

### 2.2.2 Root-MUSIC

I also evaluate the Root-MUSIC algorithm as a classical signal processing baseline. The estimator is configured using a known ULA geometry with 8 elements and a carrier frequency of 5.25 GHz. The number of signal sources is provided as a parameter, and forward-backward spatial smoothing is enabled to improve robustness in scenarios with correlated sources.

The input to MUSIC is the received complex baseband signal matrix $\mathtt{X}$ of size $8 \times T$, where $T$ is the number of temporal snapshots. The output is a set of estimated angles of arrival in radians. For evaluation, these estimates are sorted and compared against ground truth AoAs using root mean square error (RMSE). MUSIC serves as a strong traditional baseline, particularly effective at high SNR and for well-separated sources, making it suitable for evaluating the relative strengths of learning-based methods under various conditions.

### 2.2.3 Random Guessing Baseline

To establish a lower bound on estimation performance, I include a random guessing baseline. For each test sample, a set of random angles is sampled uniformly from the full angular range used in data generation, i.e., $[-60°, 60°]$. The number of guesses matches the known number of sources, and the angles are converted to radians for evaluation:

$$\hat{\theta}_k \sim \mathcal{U}(-\pi/3, \pi/3), \quad k = 1, \ldots, K. \tag{2.1}$$

These randomly sampled estimates are sorted and compared against the ground truth angles using the root mean square error (RMSE). This baseline provides a reference point for interpreting model performance in terms of meaningful error reduction over a naive, uninformed estimator. While trivial, it emphasizes the importance of learning meaningful spatial structure from the array data.

### 2.2.4 Embeddings with Fully Connected Neural Network

To address the challenges of accurate multi-source angle of arrival (AoA) estimation from raw antenna array signals, I propose a novel neural architecture that leverages a pre-trained self-supervised audio representation model—`Data2VecAudio`—as a feature extractor, followed by a compact regression head for angle prediction. The model is trained end-to-end and operates directly on time-domain complex signals from a uniform linear array (ULA).

**Model Inputs.** Each input sample is a complex-valued signal matrix of shape $(T, M)$, where $T$ is the number of temporal snapshots and $M = 8$ is the number of antenna elements. To handle complex data in real-valued neural networks, the real and imaginary parts of each antenna channel are concatenated, resulting in a $2M$-channel input sequence. The input is then flattened and normalized before being tokenized and padded using a processor compatible with the `facebook/data2vec-audio-base-960h` model. The resulting tensor has shape $(B, L)$, where $B$ is the batch size and $L$ is the padded input length.

**Architecture.** The core of the model is a `Data2VecAudioModel` initialized with the base configuration, which provides a contextual representation of the audio-like input sequence. The output of the model is a tensor of shape $(B, T', H)$, where $T'$ is the output token length and $H$ is the hidden size of the encoder. To reduce this to a fixed-size representation, I apply 1D adaptive average pooling along the time dimension, yielding a pooled embedding of shape $(B, H)$. This vector is then passed through a three-layer regression head:

1. A linear layer projecting to a hidden dimension of 256, followed by ReLU activation.
2. A second linear + ReLU block.
3. A final linear layer projecting to 2 output values, passed through a `tanh` activation.

The final output represents two AoA values in radians, scaled to the range $[-\pi/2, \pi/2]$.

Figure 2.2 shows a visual representation of the architecture described above.

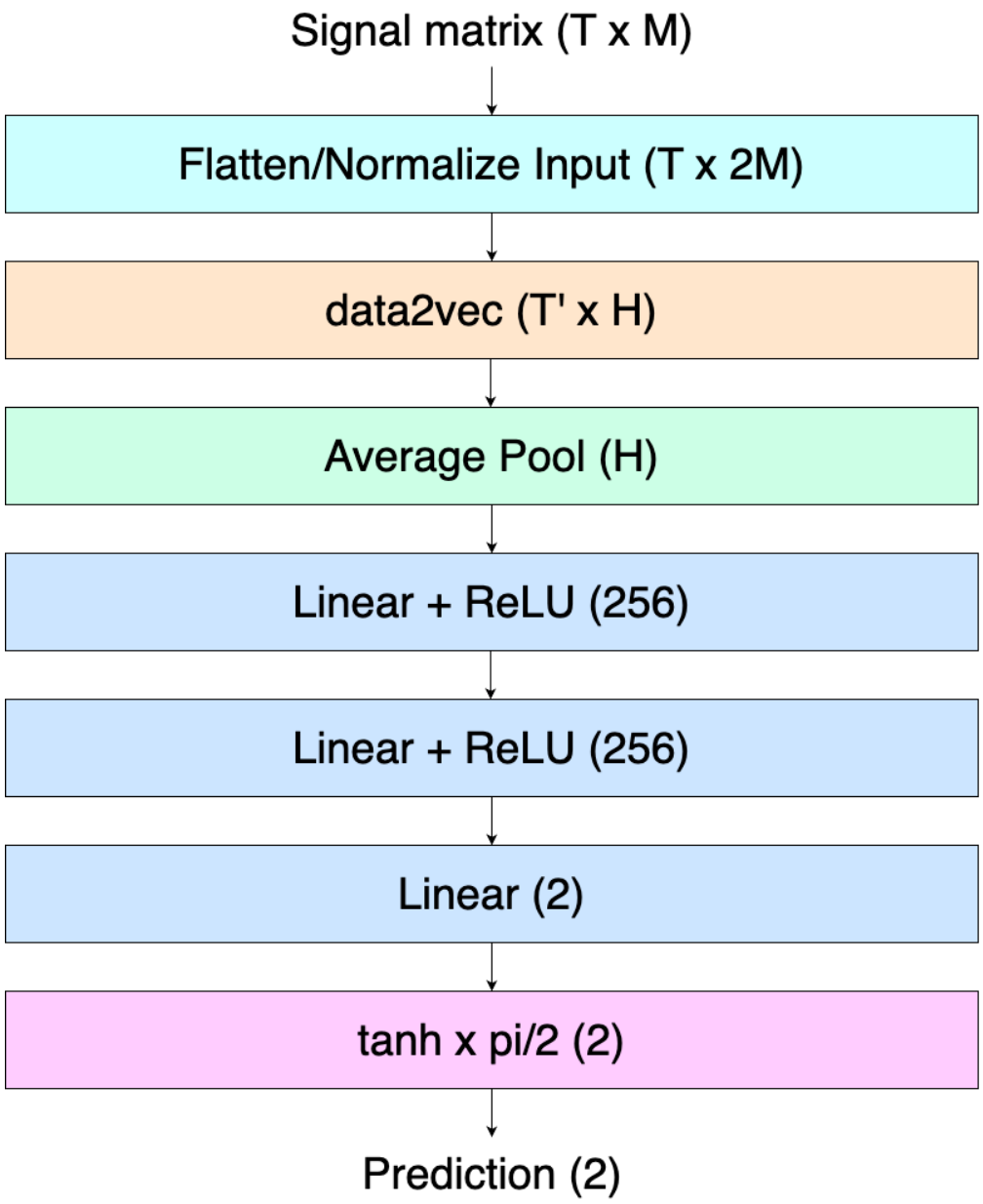


Figure 2.2: Proposed angle of arrival estimation model architecture.

**Loss Function and Evaluation.** Since the two source angles are unordered, I employ a custom permutation-invariant mean squared error (MSE) loss function. The model computes both the direct and reversed pairing between predictions and ground truth angles, and minimizes the lower of the two losses. This enables the model to remain agnostic to the order of estimated sources:

$$\mathcal{L}_{\text{unordered}} = \min\left(\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|^2, \|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^{\text{flip}}\|^2\right). \tag{2.2}$$

During evaluation, I compute the average loss over the dataset using this unordered MSE and report root mean squared error (RMSE) metrics. The design prioritizes generalization across

variable-length signals, direct operation on raw complex I/Q data, and efficient fine-tuning of self-supervised representations for spatial estimation tasks.

## 2.3 Evaluation Metrics

### 2.3.1 Root Mean Squared Error (RMSE)

The primary evaluation metric used for angle of arrival estimation is the root mean squared error (RMSE) between the estimated and ground truth angles. For a given test example with $K$ sources, let $\hat{\boldsymbol{\theta}} = [\hat{\theta}_1, \ldots, \hat{\theta}_K]$ denote the estimated angles, and $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_K]$ the ground truth angles, both expressed in radians and sorted to resolve permutation ambiguity. The RMSE is computed as:

$$\text{RMSE} = \sqrt{\frac{1}{K} \sum_{k=1}^{K} \left( \hat{\theta}_k - \theta_k \right)^2}. \tag{2.3}$$

This metric quantifies the average angular deviation per source and provides a direct measure of estimation accuracy. Lower RMSE values indicate more precise direction estimates. Note: both RMSE and RMSPE (Root Mean Squared Phase Error) are used interchangabely thorughout this paper, since the physical quantity I am evaluating for is angle, or phase.

### 2.3.2 Cumulative Distribution Function (CDF) of RMSE

To assess the distribution of estimation errors across an entire dataset, I compute and visualize the cumulative distribution function (CDF) of RMSE values. Given a set of $N$ test examples, each producing an RMSE value, the empirical CDF provides the proportion of samples with RMSE less than or equal to a given threshold $\tau$:

$$\text{CDF}(\tau) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I} \left( \text{RMSE}_i \leq \tau \right), \tag{2.4}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Plotting the CDF allows for comparison of methods in terms of their reliability and consistency: methods with steeper CDFs that rise quickly toward 1 indicate better overall performance across samples.

### 2.3.3 Standard Deviation of RMSE

In addition to mean accuracy, I report the standard deviation of RMSE values across the test set as a measure of estimator stability. Given $N$ RMSE values $\{\text{RMSE}_1, \ldots, \text{RMSE}_N\}$, the standard deviation is defined as:

$$\sigma_{\text{RMSE}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left( \text{RMSE}_i - \bar{\text{RMSE}} \right)^2}, \tag{2.5}$$

where $\bar{\text{RMSE}}$ is the mean RMSE. A lower standard deviation suggests that the estimator performs consistently across diverse conditions, while a higher value may indicate sensitivity to factors such as SNR, source separation, or snapshot count.

# Chapter 3

# Model Training

## 3.1 Software and Frameworks

This work leverages both MATLAB and Python-based deep learning frameworks to support the end-to-end pipeline for synthetic data generation, model training, and evaluation. The tools were chosen to make use of domain-specific simulation capabilities alongside modern neural network training infrastructure.

### 3.1.1 MATLAB

All training and testing datasets were synthetically generated using MATLAB R2024b. MATLAB was also used to evaluate classical signal processing algorithms such as MUSIC and Beamscan, and to compute baseline metrics for comparison with the proposed deep learning models. In addition, all plots and visualizations presented in this thesis were created using MATLAB, including histogram distributions, AoA signal visualizations, and performance figures.

The implementation relied on core MATLAB functionality as well as the Signal Processing Toolbox and Antenna Toolbox. These toolboxes provided built-in support for array signal modeling, sensor geometry configuration, and accurate signal simulation using custom antenna array definitions.

### 3.1.2 PyTorch

All deep learning model development and training were conducted using the PyTorch framework. The neural architecture proposed in this work is based on the `Data2VecAudioModel`, which was obtained from the Hugging Face Transformers library [1]. This model served as the backbone for feature extraction, and was fine-tuned end-to-end for AoA regression.

Model training was performed on a Google Colab Pro+ environment equipped with an NVIDIA A100 GPU, enabling efficient experimentation with batch training, loss function design, and hyperparameter tuning. Additional libraries such as `transformers`, `torchaudio`, and `scikit-learn` were used to support pre-processing, evaluation, and data management workflows.

## 3.2 Model Training Pipeline

To train the proposed AoA estimation model, I implemented a full training pipeline using PyTorch, starting from data loading and preprocessing, to model training, evaluation, and checkpointing. The input dataset was generated in MATLAB and saved in `.mat` format, consisting of complex-valued time-domain signals and their corresponding ground truth angle labels. I loaded this data into Python using the `scipy.io` library.

### 3.2.1 Preprocessing and Dataset Preparation

The complex antenna array signals were preprocessed by separating their real and imaginary components, which were then concatenated across the channel dimension to create a real-valued

[1] https://huggingface.co/facebook/data2vec-audio-base-960h

representation. I normalized each sample individually and flattened the data to match the input format required by the `facebook/data2vec-audio-base-960h` processor from the Hugging Face Transformers library. I defined a custom PyTorch `Dataset` and `collate_fn` to handle batching and preprocessing.

### 3.2.2 Model Training and Loss Function

The model was trained using the Adam optimizer. I used a custom permutation-invariant root-mean squared error (RMSE) loss function, which compares the predicted angles to the ground truth in both direct and flipped order, selecting the minimum loss. This loss accounts for the fact that the order of the two estimated angles does not matter. I seeded the random number generators to ensure reproducibility.

### 3.2.3 Hyperparameter Search

To determine the optimal learning rate and batch size, I conducted a grid search over learning rates {1e-5, 5e-4, 1e-4} and batch sizes {128, 256, 512}. I split 10% of the dataset into a validation set and trained each configuration for 3 epochs. After evaluating each model on the validation set, I selected the configuration with the lowest validation loss. Figure 3.1 shows the training RMSE loss over all configurations during the grid search. Likewise, Figure 3.2 shows the validation RMSE loss over all configurations during the grid search. Finally, Table 3.1 highlights results from Epoch 3 of training. In both validation and testing, a learning rate of `1e-5` produced far superior results compared to any other hyperparameter configuration, regardless of batch size. However, a batch size of `128` produced both the lowest training and validation losses out of the entire set. Therefore, a learning rate of `1e-5` and batch size of `128` were chosen for model training.

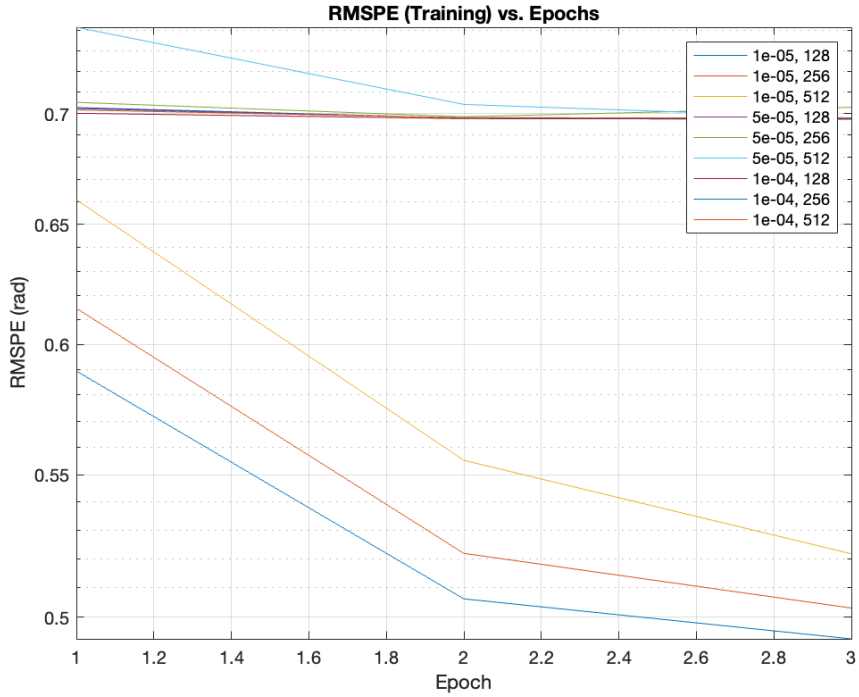


Figure 3.1: Training RMSE Loss during hyperparameter grid search.

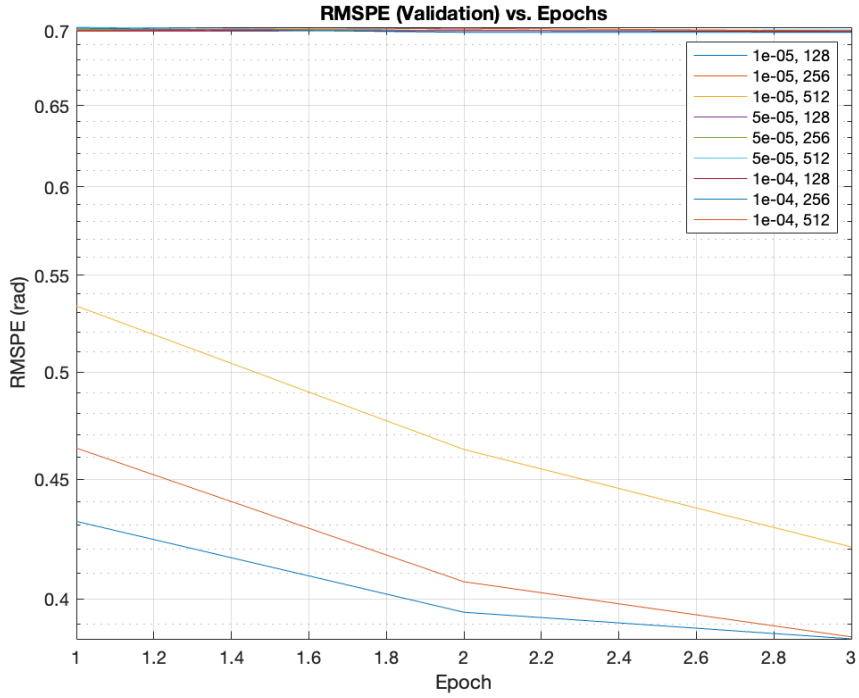


Figure 3.2: Validation RMSE Loss during hyperparameter grid search.

| Learning Rate | Batch Size | RMSE Train Loss (Epoch 3) | RMSE Validation Loss (Epoch 3) |
|---|---|---|---|
| 1e-05 | 128 | **0.4928** | **0.3845** |
| 1e-05 | 256 | 0.5031 | 0.3852 |
| 1e-05 | 512 | 0.5217 | 0.4209 |
| 5e-04 | 128 | 0.6979 | 0.6991 |
| 5e-04 | 256 | 0.7029 | 0.6994 |
| 5e-04 | 512 | 0.6979 | 0.7004 |
| 1e-04 | 128 | 0.6974 | 0.6992 |
| 1e-04 | 256 | 0.6977 | 0.6986 |
| 1e-04 | 512 | 0.6980 | 0.6997 |

Table 3.1: Hyperparameter search results at Epoch 3 across different learning rates and batch sizes. The best configuration (highlighted during evaluation) was a learning rate of `1e-5` and batch size of `128`, achieving the lowest validation loss.

### 3.2.4 Final Training

Using the best-performing hyperparameters from the grid search, I retrained the model from scratch for 20 epochs. I used a learning rate scheduler (`ReduceLROnPlateau`) to reduce the learning rate if the validation loss plateaued. After each epoch, I saved a checkpoint containing the model and optimizer state, as well as a NumPy file with the current training and validation loss. This allowed for resuming training and later analysis. Figure 3.3 highlights the training and validation RMSE over all epochs, while Table 3.2 exact RMSE values over 5 equally spaced snapshots during training.

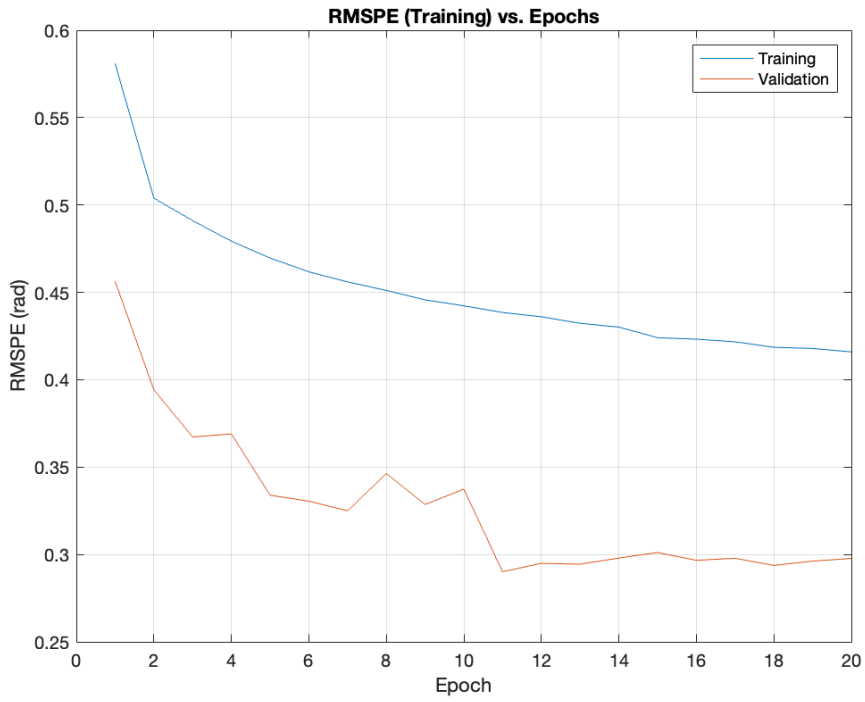


Figure 3.3: Training and validation RMSE loss over 20 epochs during model training.

| Epoch | RMSE Train Loss | RMSE Validation Loss |
|---|---|---|
| 4 | 0.4783 | 0.3690 |
| 8 | 0.4511 | 0.3463 |
| 12 | 0.4630 | 0.2949 |
| 16 | 0.4232 | 0.2967 |
| 20 | 0.4159 | 0.2978 |

Table 3.2: Training and validation RMSE loss every 4 epochs throughout model training.

Over the course of 20 epochs, the training loss steadily decreased from an initial value of 0.4783 to 0.4159, indicating consistent convergence of the model on the training data. The validation loss, however, started significantly lower at 0.3690 and quickly dropped to a minimum of 0.2901 by epoch 11, before stabilizing around 0.2960 for the remaining epochs. Notably, the validation loss remained consistently and significantly lower than the training loss throughout the entire training process.

This uncommon pattern suggests that the model generalizes exceptionally well to the validation set, potentially due to regularization effects introduced by the pre-trained `Data2VecAudio` backbone and the inherent structure of the synthetic dataset. It is also possible that the training set contains more difficult or noisy examples than the validation set, leading to higher average training error. Additionally, the use of adaptive pooling and dropout-like behavior from masked representations in `Data2VecAudio` may contribute to improved generalization performance. Overall, these results indicate that the model is not overfitting and is learning transferable features that

perform robustly across unseen data.

# Chapter 4

# Results and Discussion

## 4.1 Performance Analysis

As mentioned in 2.1.2, there were 3 test data sets generated: one that evaluates model performance on two signals coming from closely spaced sources, one that evaluates model performance on two signals with varying observation durations, and one that evaluates model performance under varying SNR. Please refer to 2.1.2, 2.1.2, and 2.1.2 for more details. This section analyzes the performance of all 4 models evaluated (MUSIC, Beamscan, the novel deep learning architecture, and random guessing). Each section contains a table with descriptive statistics on model performance for each model evaluated on the dataset, a CDF plot of RMSE, and the RMSE plotted vs the experimental variable for each experiment.

### 4.1.1 Closely-Spaced Sources

Table 4.1 contains descriptive statistics of RMSPE for the evaluated models. MUSIC significantly outperforms all other methods, achieving the lowest mean RMSPE (0.0582) and an extremely low median (0.0005), indicating high precision in closely spaced scenarios. Deep Learning also performs well, with a low mean RMSPE (0.0696) and tight quartile range, suggesting stable predictions. The results from the Deep Learning model also indicate that it performs just as well, if not better on unseen data when compared to training and validation results, indicating it is able to generalize well. Beamscan performs moderately, with higher mean, median, and standard deviation of error than either of the other two models.

Figure 4.1 paints a similar story, with MUSIC having more than 90% of predictions with less than 0.01 RMSE, the deep learning model having more than 90% of predictions under 0.1 RMSE, and Beamscan faring significantly worse. Finally, Figure 4.2 provides insight on the distribution of error across varying source distances. Both the conventional algorithms (MUSIC and Beamscan) faced significant performance degradation when faced with sources less than 0.05 radians. Meanwhile, the trained model achieved relatively stable performance across all source separations, suggesting its robustness signal source characteristics when compared to traditional signal processing approaches.

| Model | Mean | Std | Q1 | Median (Q2) | Q3 |
|---|---|---|---|---|---|
| MUSIC | 0.0582 | 0.2070 | 0.0003 | 0.0005 | 0.0012 |
| Beamscan | 0.3213 | 0.4252 | 0.0083 | 0.2537 | 0.3333 |
| Deep Learning | 0.0696 | 0.1030 | 0.0242 | 0.0404 | 0.0685 |
| Random | 0.7284 | 0.3964 | 0.4180 | 0.6884 | 0.9884 |

Table 4.1: Descriptive statistics (mean, standard deviation, and quartiles) of RMSPE for different models on the Closely Spaced Sources dataset.

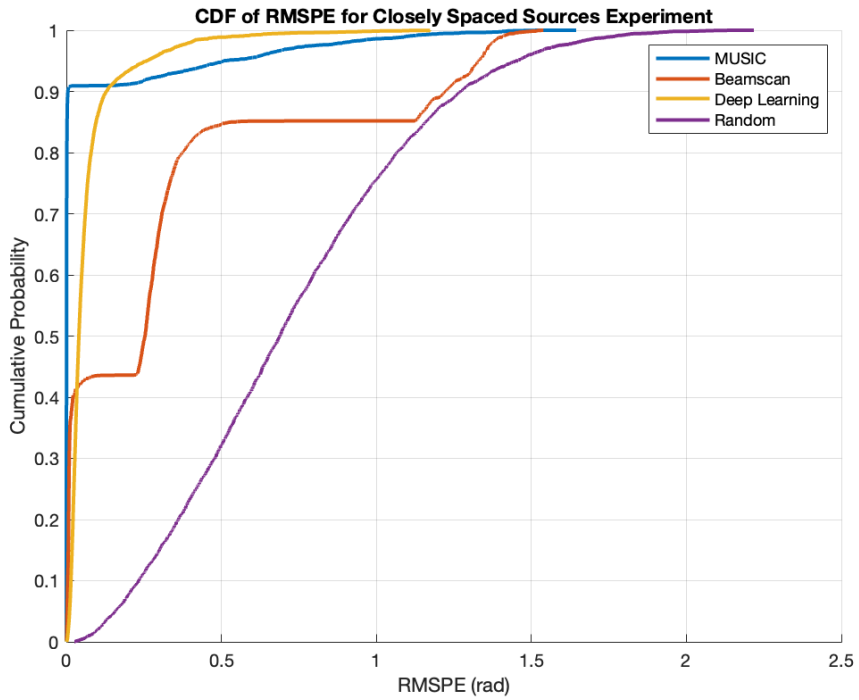


Figure 4.1: CDF of RMSE for closely spaced sources experiment.

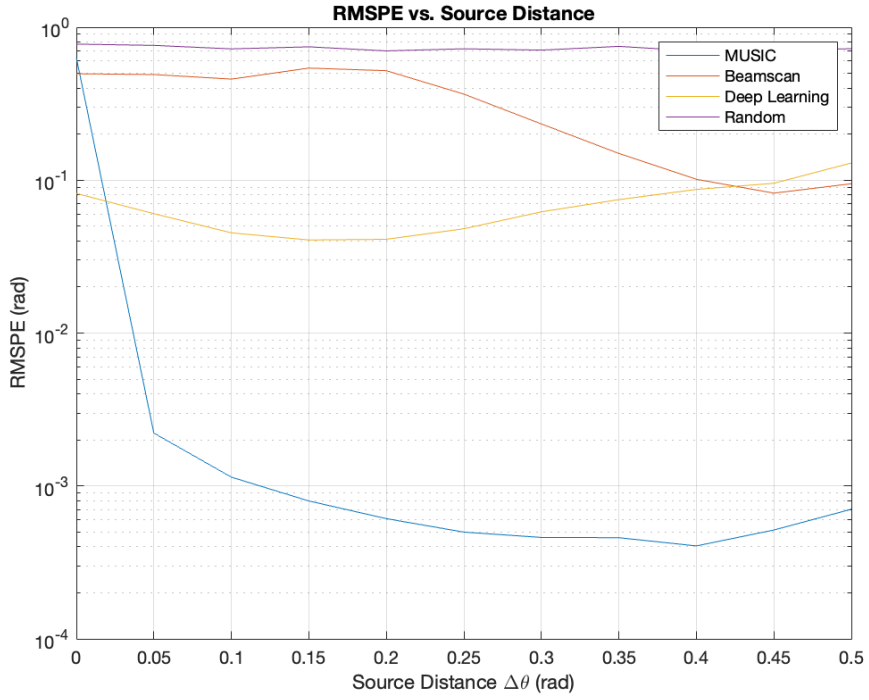


Figure 4.2: Mean RMSE per angular separation for evaluated models. Angular separations $\Delta$ between sources are selected from the range $[0, 0.5]$ radians, in increments of 0.05.

### 4.1.2 Variable Snapshot Count

Table 4.2 contains descriptive statistics of RMSPE for the evaluated models. This experiment further emphasizes MUSIC's robustness, with a low mean RMSPE (0.0046) and minimal spread across all quartiles. Deep Learning shows competitive and consistent performance, having the lowest error spread (0.0261) across all models. Beamscan exhibits a wider error distribution, and the Random baseline remains the least accurate.

Figure 4.3 and Figure 4.4 highlight all the models' ability to generate consistently accurate estimations with limited data; there is virtually no drop in performance when comparing predictions with 20 snapshots (minimum) and 200 snapshots (maximum). MUSIC, in particular is unaffected-almost all predictions are under 0.01 RMSE. The data also suggest that the other experimental factors (SNR and source angular separation) contribute more to model performance than signal length.

| Model | Mean | Std | Q1 | Median (Q2) | Q3 |
|---|---|---|---|---|---|
| MUSIC | 0.0046 | 0.0580 | 0.0003 | 0.0004 | 0.0008 |
| Beamscan | 0.1140 | 0.2766 | 0.0045 | 0.0070 | 0.0165 |
| Deep Learning | 0.0438 | 0.0261 | 0.0247 | 0.0391 | 0.0580 |
| Random | 0.6157 | 0.3338 | 0.3606 | 0.5743 | 0.8329 |

Table 4.2: Descriptive statistics (mean, standard deviation, and quartiles) of RMSPE for different models on the Variable Snapshot dataset.

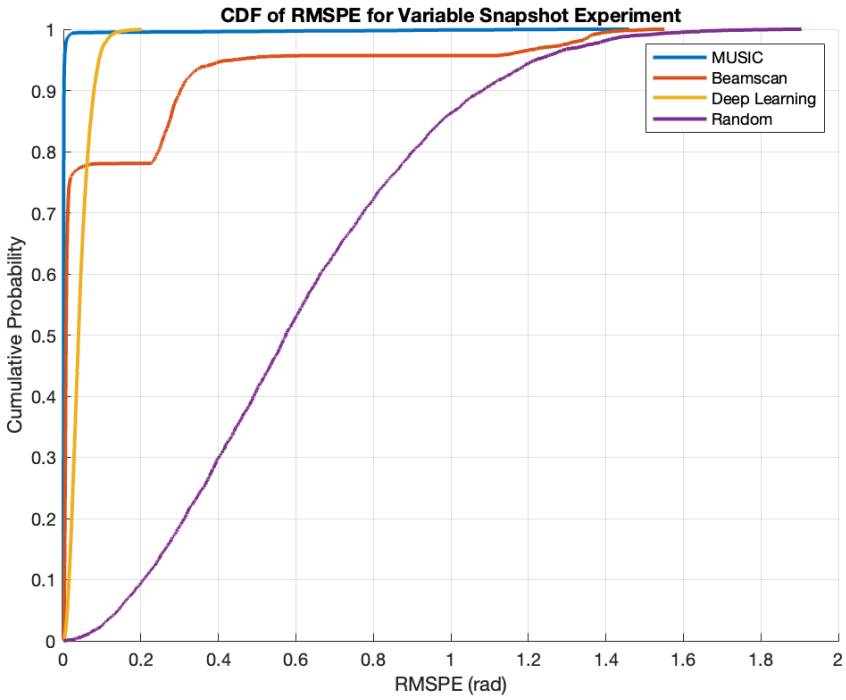


Figure 4.3: CDF of RMSE for variable snapshot experiment.

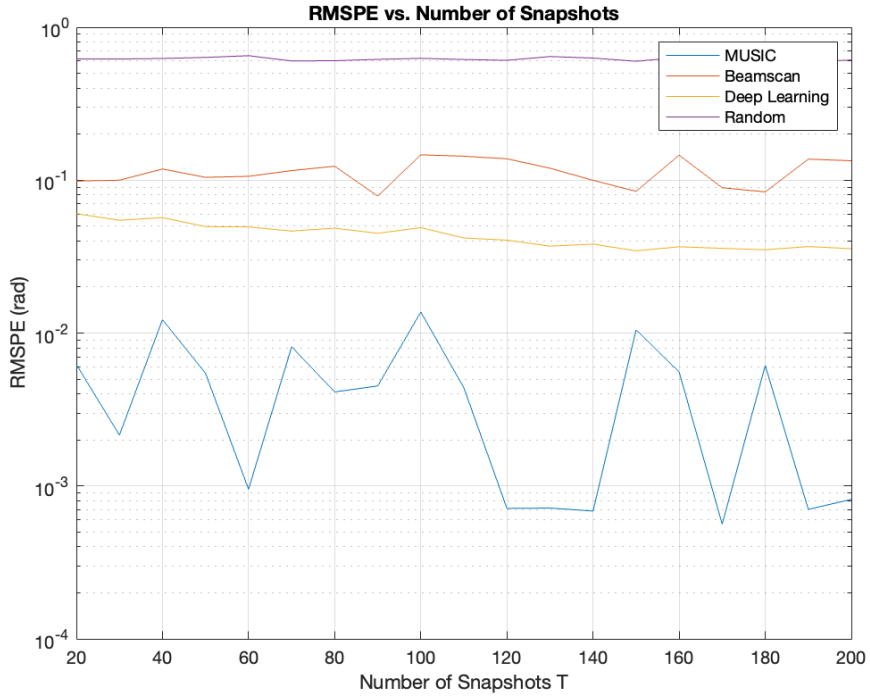


Figure 4.4: Mean RMSE per snapshot for evaluated models, where snapshots $T$ is chosen from the set $\{20, 30, \ldots, 200\}$.

### 4.1.3 Variable Signal-to-Noise Ratio

Table 4.3 contains descriptive statistics of RMSPE for the evaluated models. Under varying noise levels, MUSIC remains the top performer with a mean RMSPE of 0.1064, followed closely by Deep Learning (0.1618). Notably, Beamscan shows higher variance and a large upper quartile (Q3 = 0.2659), indicating susceptibility to high-noise conditions. The Random model again yields the worst performance. Furthermore, the mean, median, and error spread for all models is significantly higher than either of the other two experiments.

Figure 4.5 and Figure 4.6 further indicate SNR's impact on model performace; the CDF curve is less steep for all models, and the SNR plot shows that all models suffer significantly from high signal attenuation; the Deep Learning model faces significant degradation for any SNR at -5dB, while both MUSIC and Beamscan face significant degradation at -10dB.

| Model | Mean | Std | Q1 | Median (Q2) | Q3 |
|---|---|---|---|---|---|
| MUSIC | 0.1064 | 0.2587 | 0.0008 | 0.0037 | 0.0255 |
| Beamscan | 0.1776 | 0.3185 | 0.0055 | 0.0113 | 0.2659 |
| Deep Learning | 0.1618 | 0.2701 | 0.0278 | 0.0506 | 0.1417 |
| Random | 0.6115 | 0.3352 | 0.3571 | 0.5760 | 0.8214 |

Table 4.3: Descriptive statistics (mean, standard deviation, and quartiles) of RMSPE for different models on the Variable SNR dataset.

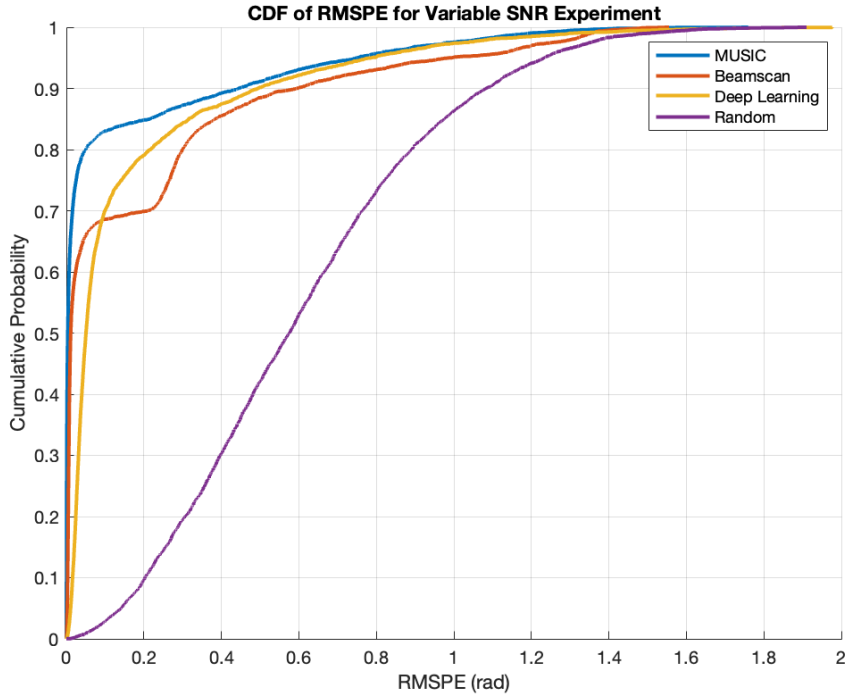


Figure 4.5: Mean RMSE for variable SNR experiment.

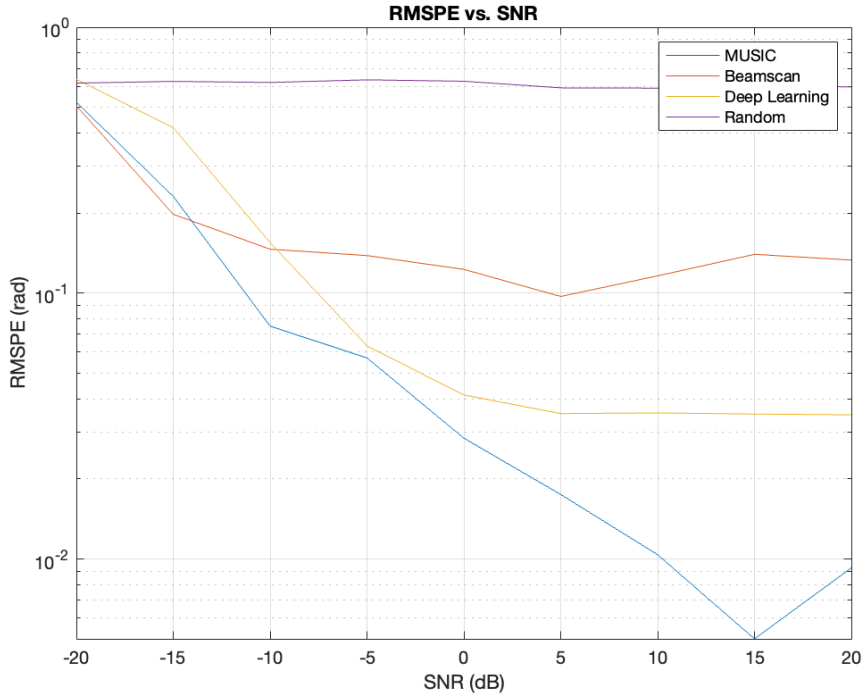


Figure 4.6: Mean RMSE per SNR value for evaluated models, where SNR is selected from the set $\{-20, -15, \ldots, 20\}$ dB.

## 4.2 Limitations and Future Work

The results presented in this work demonstrate that a deep learning-based model leveraging pre-trained vector embeddings can perform comparably to classical signal processing algorithms for the task of angle of arrival (AoA) estimation. Although the proposed model does not outperform the MUSIC algorithm—which remains the state-of-the-art in subspace-based AoA estimation—it shows promise in using high-level representations from pre-trained models for signal processing tasks.

Notably, the experiment illustrates the generalizability of current self-supervised embedding models: `Data2VecAudio`, originally trained for vision, speech, and language tasks, was effectively repurposed as a feature extractor for wireless array signal data. The model successfully captured spatio-temporal patterns inherent in complex-valued antenna signals, even in the absence of domain-specific fine-tuning. Additionally, the model was able to handle inputs of variable temporal lengths, suggesting robustness and consistent performance across varying data availability and quality.

Despite these encouraging findings, several limitations must be acknowledged:

1. **Synthetic Dataset.** All training and evaluation data were generated in simulation. Real-world Wi-Fi signal data is considerably noisier and typically not formatted for direct input to learning models. Deploying the proposed approach on real systems would require extensive signal preprocessing, calibration, and possibly hardware synchronization.

2. **Uncorrelated Noise Assumption.** The simulation environment assumed uncorrelated additive noise across sources. However, in practical scenarios, multipath propagation—caused by room geometry, reflections, and obstructions—results in correlated interference, delayed arrivals, and phase-amplitude distortions that were not accounted for during training.

3. **Fixed Number of Sources.** The current model was trained and evaluated on scenarios involving exactly two signal sources. Real-world deployments often encounter a dynamic number of interfering sources. It remains unclear whether the model generalizes to more complex environments involving a higher number of emitters.

4. **Input Representation.** The architecture used a simplified preprocessing scheme that flattened all spatial channels before feeding the signal into the embedding model. This approach may discard critical spatial dependencies. More sophisticated encoding methods that preserve spatial and temporal structure could lead to improved performance.

5. **Resource Constraints.** Model training required substantial computational resources, including access to data center-grade GPUs. However, many applications of AoA estimation—such as in edge computing or embedded systems—have strict latency and compute constraints. The current architecture is unlikely to meet such requirements without further optimization.

Future work should aim to address these limitations to improve the model's efficiency, scalability, and practical utility. Specifically, future efforts should focus on evaluating the model on real-world data, incorporating multipath-aware data generation, enabling support for variable and

unknown numbers of sources, and exploring architecture designs that better preserve the spatial-temporal structure of antenna array data. Finally, reducing model complexity and latency will be essential for deploying such systems in practical, real-time environments.

# Bibliography

[1] AHMED, A. U., ARABLOUEI, R., DE HOOG, F., KUSY, B., JURDAK, R., AND BERGMANN, N. Estimating angle-of-arrival and time-of-flight for multipath components using wifi channel state information. *Sensors 18*, 6 (2018).

[2] BAEVSKI, A., BABU, A., HSU, W.-N., AND AULI, M. Efficient self-supervised learning with contextualized target representations for vision, speech and language, 2022.

[3] BARTHELME, A., AND UTSCHICK, W. Doa estimation using neural network-based covariance matrix reconstruction. *IEEE Signal Processing Letters 28* (2021), 783–787.

[4] ELBIR, A. M. Deepmusic: Multiple signal classification via deep learning. *IEEE Sensors Letters 4*, 4 (2020), 1–4.

[5] GRUMIAUX, P.-A., KITIĆ, S., GIRIN, L., AND GUÉRIN, A. A survey of sound source localization with deep learning methods. *The Journal of the Acoustical Society of America 152*, 1 (07 2022), 107–151.

[6] KOTARU, M., JOSHI, K., BHARADIA, D., AND KATTI, S. Spotfi: Decimeter level localization using wifi. *SIGCOMM Comput. Commun. Rev. 45*, 4 (Aug. 2015), 269–282.

[7] LIU, J., WANG, T., LI, Y., LI, C., WANG, Y., AND SHEN, Y. A transformer-based signal denoising network for aoa estimation in nlos environments. *IEEE Communications Letters 26*, 10 (2022), 2336–2339.

[8] LIU, Z.-M., ZHANG, C., AND YU, P. S. Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections. *IEEE Transactions on Antennas and Propagation 66*, 12 (2018), 7315–7327.

[9] MERKOFER, J. P., REVACH, G., SHLEZINGER, N., ROUTTENBERG, T., AND VAN SLOUN, R. J. G. Da-music: Data-driven doa estimation via deep augmented music algorithm. *IEEE Transactions on Vehicular Technology 73*, 2 (2024), 2771–2785.

[10] MIKOLOV, T., CHEN, K., CORRADO, G., AND DEAN, J. Efficient estimation of word representations in vector space, 2013.

[11] MOLAEI, A. M., ZAKERI, B., ANDARGOLI, S. M. H., ABBASI, M. A. B., FUSCO, V., AND YURDUSEVEN, O. A comprehensive review of direction-of-arrival estimation and localization approaches in mixed-field sources scenario. *IEEE Access 12* (2024), 65883–65918.

[12] PAPAGEORGIOU, G. K., SELLATHURAI, M., AND ELDAR, Y. C. Deep networks for direction-of-arrival estimation in low snr. *IEEE Transactions on Signal Processing 69* (2021), 3714–3729.

[13] ROY, P., AND CHOWDHURY, C. A survey on ubiquitous WiFi-based indoor localization system for smartphone users from implementation perspectives. *CCF Trans. Pervasive Comp. Interact. 4*, 3 (Sept. 2022), 298–318.

[14] ROY, R., AND KAILATH, T. Esprit-estimation of signal parameters via rotational invariance techniques. *IEEE Transactions on Acoustics, Speech, and Signal Processing 37*, 7 (1989), 984–995.

[15] SCHMIDT, R. Multiple emitter location and signal parameter estimation. *IEEE Transactions on Antennas and Propagation 34*, 3 (1986), 276–280.

[16] VAN VEEN, B., AND BUCKLEY, K. Beamforming: a versatile approach to spatial filtering. *IEEE ASSP Magazine 5*, 2 (1988), 4–24.

[17] WU, L., LIU, Z.-M., AND HUANG, Z.-T. Deep convolution network for direction of arrival estimation with sparse prior. *IEEE Signal Processing Letters 26*, 11 (2019), 1688–1692.

[18] XIONG, J., AND JAMIESON, K. ArrayTrack: A Fine-Grained indoor location system. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)* (Lombard, IL, Apr. 2013), USENIX Association, pp. 71–84.