



Universal Beamforming: A Deep RFML Approach

Hai N. Nguyen
nguyen.hai@northeastern.edu
Khoury College of Computer Sciences
Northeastern University

Guevara Noubir
g.noubir@northeastern.edu
Khoury College of Computer Sciences
Northeastern University

ABSTRACT

We introduce, design, and evaluate a set of universal receiver beamforming techniques. Our approach and system DEFORM, a Deep Learning (DL)-based RX beamforming achieves significant gain for multi-antenna RF receivers while being agnostic to the transmitted signal features (e.g., modulation or bandwidth). It is well known that combining coherent RF signals from multiple antennas results in a beamforming gain proportional to the number of receiving elements. However in practice, this approach heavily relies on explicit channel estimation techniques, which are link specific and require significant communication overhead to be transmitted to the receiver. DEFORM addresses this challenge by leveraging Convolutional Neural Network to estimate the channel characteristics in particular the relative phase to antenna elements. It is specifically designed to address the unique features of wireless signals complex samples, such as the ambiguous 2π phase discontinuity and the high sensitivity of the link Bit Error Rate. The channel prediction is subsequently used in the Maximum Ratio Combining algorithm to achieve an optimal combination of the received signals. While being trained on a fixed, basic RF settings, we show that DEFORM's DL model is universal, achieving up to 3 dB of SNR gain for a two-antenna receiver in extensive evaluation demonstrating various settings of modulations and bandwidths.

CCS CONCEPTS

• **Networks** → **Wireless access networks**; • **Computing methodologies** → **Neural networks**.

KEYWORDS

Universal RX beamforming; deep learning; RFML

ACM Reference Format:

Hai N. Nguyen and Guevara Noubir. 2022. Universal Beamforming: A Deep RFML Approach. In *Proceedings of the International Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM '22)*, October 24–28, 2022, Montreal, QC, Canada. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3551659.3559041>

1 INTRODUCTION

The success of wireless communications was accompanied by a dramatic crowding of the RF spectrum. Beamforming is a widely

used spatial filtering technique to steer RF emissions towards/from other devices. It enables array and diversity gains of multiple-input-single-output (MISO) systems [7, 27], and canceling interference from unwanted sources. Beamforming was extensively investigated in a variety of applications such as in radars, sonars, acoustics, astronomy, and even medical devices design [18]. Beamforming and more advanced MIMO techniques are widely used in systems targeting high throughput and spectral efficiency such as cellular systems since the third generation 3GPP 3G, and IEEE 802.11n.

Extensive research was conducted on beamforming [4, 19, 28, 30, 34]. However, beamforming in today's systems requires explicit mechanisms such as sounding and feedback in 802.11, Demodulation Reference Signal (DMRS) in 5G, training sequences, etc. This approach introduces critical limitations: Firstly, it results in additional overhead to transmit reference signals and estimate the channel. Secondly, accurate channel estimation typically exhibits long delay that is undesirable in fast-changing channels. Thirdly, it requires compatibility between the transmitter and receiver (to agree on when, what, and how reference signals are transmitted).

Advances in Deep Learning (DL) techniques and models have recently achieved great success in numerous areas such as computer vision [8], speech recognition [13], and wireless communications [21]. A deep neural network can analyze complex patterns in raw I/Q data collected from the RF front-end and accurately predict channel characteristics in a very short amount of time. Such capabilities eliminate the requirement of TX-RX compatibility and are crucial for a *universal beamforming* component that support the communications of different technologies operating over the same spectrum. However, despite of the great potentials, most of prior work on DL-based RX beamforming [16, 32] is limited to analytical and simulation results, and lack the goals of universality.

In this work, we introduce a set of techniques, and their extensive experimental evaluations demonstrating the practicality of universal beamforming for wireless receivers using Deep Learning. Our approach and system denoted DEFORM, is agnostic to the specifics of transmitted signals such as modulation, bandwidth or standard. DEFORM is designed around a deep convolutional neural network (CNN) augmented with a Maximum Ratio Combiner (MRC). It is specifically designed to address the unique features of wireless signals complex samples, such as the 2π phase ambiguous discontinuity. Also, RF links targeting low Bit Error Rates (e.g., below 10^{-4}) are sensitive to the typical variations and outliers in the continuous-valued estimations of neural networks [3]. DEFORM addresses these challenges successfully achieving the *optimal* beamforming gain. Our contributions can be summarized:

- A design, model, and algorithm for a novel Deep Learning-based *universal* receiver beamformer (DEFORM). To the best of our knowledge, our work is the first in the literature that leverages DL to enable multi-antenna receivers for beamforming RF signals

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

MSWiM '22, October 24–28, 2022, Montreal, QC, Canada

© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9479-6/22/10...\$15.00
<https://doi.org/10.1145/3551659.3559041>

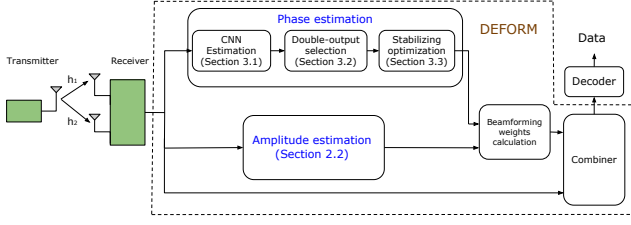


Figure 1: DEFORM's beamforming workflow

irrespective of modulations or bandwidths (and not requiring any explicit mechanisms such as sounding).

- A two-antenna SDR [1] receiver prototype leveraging DEFORM that supports arbitrary and unseen modulations and bandwidths. DEFORM is trained on a fixed basic RF settings (BPSK, (1 MHz) bandwidth, in a cable environment) and bolstered with efficient RF dataset collection process with phased augmentation.
- DEFORM is extensively evaluated (B/Q/8-PSK, 16-QAM, GMSK with different bandwidths) in cabled setup with emulated fading channel effects. DEFORM achieved the optimal 3 dB gain of a two antenna receiver.

2 PROBLEM AND APPROACH

Receiver (RX) beamforming aims to optimally combine the received signals from multiple antennas to maximize the Signal-to-Noise Ratio (SNR). In this section, we identify and formulate the RX beamforming problem and present the key ideas of DEFORM.

2.1 Theory and Challenge

Consider a transmitter and a receiver communicating through a slow fading channel. The receiver has N antenna elements, where we model the signal R_i received by antenna i consisting of the transmitted signal S adjusted by the channel gain h_i and the additive Gaussian noise N_i :

$$R_i = h_i S + N_i = s_i + N_i \quad (1)$$

Receiver beamforming aims to leverage the diversity of the independent wireless channels between the transmitting and receiving antennas by combining the receiving branches with the adequate complex *beamforming weights*:

$$R_\Sigma = \sum_{i=1}^N a_i R_i = \sum_{i=1}^N (a_i s_i + a_i N_i) \quad (2)$$

The *beamforming weights* a_i are chosen to maximize the combining Signal-to-Noise Ratio (SNR), which is given by:

$$SNR_\Sigma = \frac{(\sum_{i=1}^N a_i s_i)^2}{N_0 \sum_{i=1}^N |a_i|^2} \quad (3)$$

where we assume the noise in each branch is independent and has the same Power Spectral Density (PSD) N_0 . The maximization of SNR_Σ is solved using the Cauchy-Schwartz inequality [7] and yields the optimal weights:

$$\hat{a}_i = \frac{s_i^*}{\sum_{j=1}^N |s_j|} \quad \forall i \in 1, \dots, N \quad (4)$$

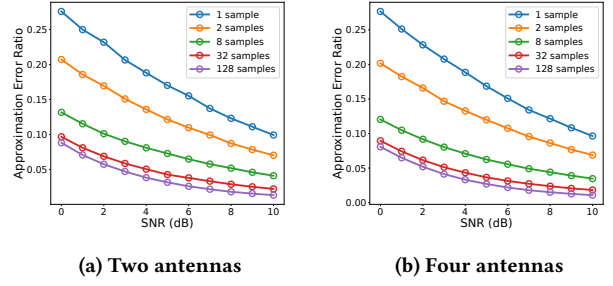


Figure 2: Error analysis of the approximation of A_i (Equation (7)) with regards to different SNR levels, number of samples, and number of antennas. Each data point is acquired by 10,000 Monte Carlo simulations in the AWGN channel.

where the denominator is the scaling factor for the weights. Substitute to Equation (3), we now have the combining SNR:

$$SNR_\Sigma = \frac{\sum_{i=1}^N s_i^2}{N_0} = \sum_{i=1}^N SNR_i \quad (5)$$

The optimal combining SNR is the sum of the SNRs from all receiving branches. In the best case scenario when all branches have the same SNRs, this beamforming technique (also known as Maximal-Ratio Combining [7]) can achieve a final SNR of N times the SNR acquired from a single branch. For instance, with a two-antenna receiver we gain twice the SNR (corresponding to a 3 dB gain).

The main challenge to achieving the optimal combiner is how to estimate the optimal weight $\hat{a}_i$ for each receiving branch i . In Equation (4), $\hat{a}_i$ is dependent on s_i ($s_i = h_i S$), which is unknown to the receiver. Conventional techniques, which are still extensively used in OFDM or MIMO systems [26], have to insert mutually-known data such as a training sequence or pilot symbols into the transmitted signals that causes significant communications overhead. Meanwhile, phased-array systems [27] do not require channel estimation, but are especially inaccurate against multi-path effects which add up the channel gains from multiple paths and distort the channel characteristics of the direct path.

2.2 Approach

We design the receiver beamforming system with the goal to estimate the optimal beamforming weights $\hat{a}_i$ accurately. We first reformulate $\hat{a}_i$ from Equation (4) in the polar representation:

$$\hat{a}_i = \frac{|s_i|}{\sum_{j=1}^N |s_j|} e^{-j\theta_i} \quad \forall i \in 1, \dots, N \quad (6)$$

To find $\hat{a}_i$, we need to estimate the amplitude $A_i = \frac{|s_i|}{\sum_{j=1}^N |s_j|}$ and the phase θ_i . First, we present a simple approach to estimate the amplitude. Out of N receiving branches, we pick an arbitrary branch k , and approximate A_i for every branch i with the transformation:

$$A_i = \frac{\frac{|s_i|}{|s_k|}}{\sum_{j=1}^N \frac{|s_j|}{|s_k|}} \approx \frac{\frac{|R_i|}{|R_k|}}{\sum_{j=1}^N \frac{|R_j|}{|R_k|}} \quad (7)$$

where the ratio $\frac{|s_i|}{|s_k|}$ is approximated by the amplitude ratio of received signals $\frac{|R_i|}{|R_k|}$ for every $i \neq k$ (If $i = k$, the ratio is 1). It is easy to see that the approximation is correct when $|s_i| \approx |s_k| \forall i \in 1, \dots, N$. Nonetheless, as $|s_i|$ gets significantly bigger than $|s_k|$, the approximation error increases. The worst case scenario is when $|s_i| \gg |s_k|$ and at the same time, the signal power $|s_k|^2$ is close to the noise PSD N_0 . However, we can minimize the approximation error by calculating the average of A_i over multiple RF samples, instead of only using a single sample. As shown by the numerical analysis in Figure 2, the approximation error decreases as we increase the number of samples used for the approximation, regardless the number of antennas in the system. When 128 samples are used for the estimation, we can reduce the error to less than 5% even at a very low SNR (i.e., about 3dB in Figure 2). Moreover, many wireless communications typically requires a sufficiently high SNR to operate (e.g., over 20 dB for Wi-Fi [5]). As we will discuss in later sections, a SNR of 10 dB is the minimum requirement for the RX to achieve a target Bit Error Rate of less than 10^{-4} .

Estimating the phase θ_i is more challenging, especially in the absence of explicit information from the transmitter (e.g. reference signal or sounding mechanism). This is due to the effect of multi-path propagation, in which the constructive and destructive phase combining of multiple copies of the signal traversing the space is typically unpredictable. At the same time, estimating the phase is a critical requirement for the optimal beamforming weights to make the receiving branches **co-phased** in the combined signal. Without this, the branch signals will not add up coherently and the combining signal will experience even further fading (similar to the behavior of multi-path) [7]. To address this, we present a new approach to achieve co-phasing and optimal beamforming weights. Instead of finding the **absolute** signal phase θ_i , we estimate the **relative** signal phase $\Delta\theta_i = \theta_i - \theta_k$ between the current branch i and a pre-selected arbitrary branch k , resulting in the new weight:

$$\bar{a}_i = A_i e^{-j\Delta\theta_i} \quad (8)$$

which makes the received signal from branch i co-phased with the signal from branch k . As a result, all branch signals are co-phased in the combiner and we achieve the optimal SNR gain.

To estimate the relative phase, we need to detach the signal component from the noise component. Phased-array systems [27] have been known for the capability to disentangle different components in the received signals. However, they typically perform poorly in the presence of multi-path fading [23]. To address this problem, we propose a novel Deep Learning-based universal RX beamforming system (DEFORM) that centers around an efficient, powerful Convolutional Neural Network (CNN). Inspired by the capability of CNN to filter and extract relevant low-level features from data in various areas such as text [13], RF [21], visual [25], or speech [2], we develop a CNN model that precisely estimates the relative phase directly from the branch signals. The CNN is facilitated with optimization techniques to address the unique features of wireless signals complex samples, such as the misleading 2π phase discontinuity and the link Bit Error Rate sensitivity, bolstering DEFORM's universality. DEFORM is implemented for a two-antenna receiver, and extensively evaluated in various RF settings of modulations and bandwidths to validate the universality in achieving

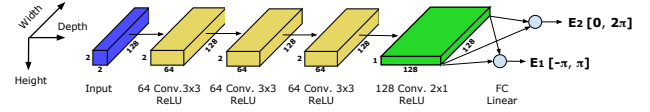


Figure 3: The CNN structure for DEFORM with rotational double-output feature.

the *optimal* beamforming gain. Here, we emphasize that existing phase-array-based systems addressing multi-path typically require more antennas [31] or multiple types of sensing hardware [15]. Furthermore, they are limited by some assumptions on the communication channels, and therefore not universal. The operation workflow of DEFORM is depicted in Figure 1, where the branch signals are combined using the optimal beamforming weights, resulting in the output signal which is sent to the decoder to decode the data. The design and optimization of the most important module - *phase estimation*, will be presented in details in Section 3.

3 PHASE ESTIMATION FOR UNIVERSAL RX BEAMFORMING

In this section, we present the design of DEFORM's phase estimation module, which leverages a CNN to accurately estimate the relative phases (which is critical for DEFORM to calculate the beamforming weights), supported by optimization techniques specially designed to address the unique features of wireless signals complex samples. We have considered several neural network architectures and choose CNN because it is very powerful to extract relevant low-level features embedded in data of various types [2, 13, 21, 25]. It is natural to leverage such capabilities to disentangle the signal components and provide accurate phase estimations.

3.1 Neural Network Design

Goals. We define three goals for the design of the CNN model. Firstly, the model should be able to process the continuous stream of I/Q data efficiently. Feeding a very long stream of samples to the CNN would significantly increase the size of the network as well as the computation cost. Secondly, the model should output the real-valued relative phase with the smallest estimation error as possible. Finally, the network estimation needs to be fast and computationally efficient. We note that there is a natural trade-off between the second and third goals, therefore we aim to find the best model that achieves optimal accuracy and speed throughout the training and validation processes.

CNN Architecture. To reduce the computational cost of the network, a long stream of RF samples is divided into equal chunks of M samples ($M = 128$ in our implementation). As complex RF samples are composed of In-phase and Quadrature components, it is intuitive to view a block of M samples as a matrix of size $2 \times M$ where each entry is an In-phase (1^{st} row) or Quadrature (2^{nd} row) value. The matrices of N antennas ($N = 2$ in our current system) are stacked along the third dimension, which consequently form a $2 \times M \times N$ tensor of real-valued elements for each input data.

After having done a network structure search, we converge on an optimized structure that achieves good performance in both

processing speed and estimation correctness. We find that as the network gets deeper, the estimation error generally decreases. However, after reaching a certain depth, the improvement becomes less significant. As one of our goals is estimation efficiency, we choose the fastest model that can achieve comparable performance with deeper models. We justify our selection by comparing with popular DL architectures (which is discussed in Section 4.1). The structure of our neural network is shown in Figure 3. Preceding layers are three convolutional layers with kernel size of 3×3 , followed by one 2×1 convolutional layer and the fully-connected output layer. 3×3 convolutional layers have been widely used in state-of-the-art DL architectures [9, 25] due to their capabilities of extracting low-level features appearing in local regions of the input data. On the other hand, the 2×1 convolutional layer aims to provide high-level semantics of angular distance with the sample-wise combining of I/Q channels. The fully connected layer synthesizes the output from previous layers and makes prediction. Rectified Linear Unit (ReLU) activation is used for the convolutional layers because it is computationally efficient and more effective against the vanishing gradient problem [6], while linear activation is applied to the output layer. We utilize Batch Normalization [12] for all convolutional layers to improve the training convergence and eliminate the needs for regularization. To avoid overfitting, we use Learning Rate Decay [33] which lowers the learning rate if the validation error remains unimproved for a period of time (e.g., a few epochs).

3.2 Rotational Double-Output

Theoretically, the neural network is required to provide one estimation for one relative phase between two receiving branches. Nonetheless, while investigating various models, we find that the phase estimation exhibits abrupt variations as the relative phase gets very close to the boundaries of phase values (i.e. upper and lower bounds of the ranges $[-\pi, \pi]$ or $[0, 2\pi]$). More specifically, the behavior is described as follows:

- If the network estimates the phase in $[-\pi, \pi]$, then the estimation abruptly fluctuates when the true value is either in $[-\pi, -(\pi - \epsilon)]$ or $[\pi - \epsilon, \pi]$.
- If the network estimates the phase in $[0, 2\pi]$, then the estimation abruptly fluctuates when the true value is either in $[0, \epsilon]$ or $[2\pi - \epsilon, 2\pi]$.

where $\epsilon \approx 0.2\pi$ based on our investigation during the validation process. We believe that such behavior is related to the discontinuity of the phase when it travels beyond the boundaries, for example, above π and below $-\pi$ for $[-\pi, \pi]$. Because of the rotational characteristics (i.e., $2\pi + \theta = \theta \pmod{2\pi}$), the phase will be shifted backward by an angle of 2π . This confuses the estimation model because those values are far apart (by a distance of 2π) in the numerical axis.

To overcome this problem, we enhance the CNN model with what we call a *rotational double-output* feature, which incorporates two estimation outputs E_1, E_2 (as depicted in Figure 3) for the relative phase converted in $[-\pi, \pi]$ and $[0, 2\pi]$, respectively. We note that the two estimations do not experience abrupt variations simultaneously. Therefore, if we know which output is experiencing errors and not usable, we can select the other output. The double-output selection is described in Algorithm 1, where we define $SELECT_1$ and

Algorithm 1: Double-output selection

Data: $E_1, E_2, STD_1, STD_2, \epsilon$
Result: E_{cur}
if $\pi - \epsilon < E_2 < \pi + \epsilon$ **then**
 | $SELECT_2 \leftarrow True$
else
 | $SELECT_2 \leftarrow False$
end if
if $-\epsilon < E_1 < \epsilon$ **then**
 | $SELECT_1 \leftarrow True$
else
 | $SELECT_1 \leftarrow False$
end if
Convert E_2 to $[-\pi, +\pi]$ by Equation (9)
if $SELECT_2 = True$ **and** $SELECT_1 = False$ **then**
 | $E_{cur} \leftarrow E_2$
else if $SELECT_2 = False$ **and** $SELECT_1 = True$ **then**
 | $E_{cur} \leftarrow E_1$
else if $SELECT_2 = True$ **and** $SELECT_1 = True$ **then**
 | **if** $STD_1 < STD_2$ **then**
 | $E_{cur} \leftarrow E_1$
 | **else**
 | $E_{cur} \leftarrow E_2$
 | **end if**
else
 | $E_{cur} \leftarrow (E_1 + E_2)/2$
end if

$SELECT_2$ to imply the correctness of the respective outputs E_1 and E_2 . If both indicators are *True*, the algorithm selects the predictions with a smaller Standard Deviation (STD) in a history windows of $K = 10$ elements. In the last case, the average value of two outputs is taken. We note that eventually, E_2 should be converted to $[-\pi, \pi]$ to avoid inconsistencies of the selections:

$$E_2 = \begin{cases} E_2 - 2\pi & \bar{E}_2 > \pi \\ E_2 & \text{else} \end{cases} \quad (9)$$

where $\bar{E}_2$ is the mean of $K = 10$ last predictions in $[0, 2\pi]$. With this feature, our CNN is trained to minimize the modified Mean Squared Error:

$$\mathcal{L}_\theta = (\Delta_{\theta_1} - E_1)^2 + (\Delta_{\theta_2} - E_2)^2 \quad (10)$$

where Δ_{θ_1} and Δ_{θ_2} are the conversions of the true relative phase Δ_θ in $[-\pi, \pi]$ and $[0, 2\pi]$, respectively.

3.3 Stabilizing The Estimations

When DEFORM system directly uses the continuous-valued phase estimations to compute the beamforming weights, it becomes susceptible to variations and outliers as typically seen in neural network models [8]. Meanwhile, practical wireless communication systems and standards require not only high precision, but also stability in the estimations as they target a Bit Error Rate in the orders of 10^{-4} for a proper operation. Short-term changes on some samples have significant and lasting impacts on the whole packet decoding and easily aggravate the Bit Error Rate. To address this, we propose two different methods for stabilizing the phase estimations:

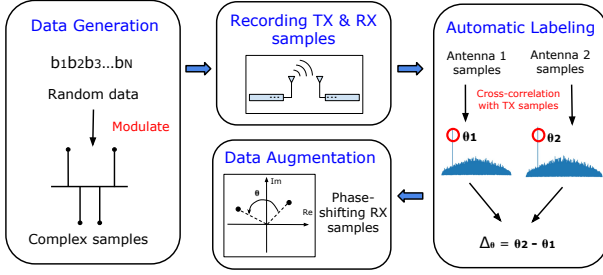


Figure 4: The procedure of building dataset for training the DL model of DEFORM.

Temporal smoothing. Because wireless channels typically change with at a much slower rate than the incoming rate of RF samples (128 RF samples collected at 1Msamp/s is only 0.128 ms long), we can stabilize the prediction at a given instant by combining with estimations from the recent past. We stabilize the estimation and improve the robustness of the RX beamforming by using the exponential smoothing function:

$$E^T = E_{cur}\lambda + E^{T-1}(1 - \lambda) \quad (11)$$

where the final phase estimation at the current time period T is computed using the estimation from previous period $T - 1$ and the current CNN output estimation acquired from Algorithm 1. Parameter λ controls the smoothness of the result, and is chosen with the best value $\lambda = 0.2$ through the validation process. It should be noted that if the offset between E^{T-1} and E_{cur} is significant (exceeding a certain threshold α , i.e. $\alpha = 1.5\pi$, we should select the instantaneous estimation E_{cur} instead. This behavior is typically seen when the channel changes from being vacant to being occupied by the transmitter. In this case, a drastic change of the phase estimation indicates the beginning of a packet that we should account for.

Algorithm 2: Multi-trial averaging

Data: N , RF samples in period T

Result: E^T

repeat N times

 Artificially adjust the phases with a random θ_{rand} ;

 Compute instantaneous E_{cur} using Algorithm 1;

 Update E_{cur} to original value with θ_{rand} ;

 Categorize E_{cur} into two clusters C_1, C_2 ;

end

Count cluster elements $COUNT_{C_1}, COUNT_{C_2}$;

Calculate average estimations $E_{C_1}^T, E_{C_2}^T$;

if $COUNT_{C_1} > COUNT_{C_2}$ **then**

$E^T \leftarrow E_{C_1}^T$

else

$E^T \leftarrow E_{C_2}^T$

end if

Multi-trial averaging. Temporal smoothing requires multiple chunks of RF samples to achieve a stabilized prediction. In cases

when the number of samples is limited but higher computation power is available (e.g., offline processing), we can instead stabilize the prediction by processing the same samples multiple times (thus *multi-trials*). The algorithm is described in Algorithm 2. To avoid repeatedly getting the same estimation, in each trial, we augment the RF samples by artificially adjusting the phases with a random θ_{rand} . We note that after being processed by the CNN, the current estimation E_{cur} needs to be re-adjusted to account for the prior augmentation. To deal with possible drastic changes, we categorizes the estimations across N trials into two clusters based on their values: We maintain the average estimations $E_{C_1}^T$ and $E_{C_2}^T$ for the clusters at time period T . An estimation E_{cur} for any trial is categorized into cluster C_1 if $|E_{cur} - E_{C_1}^T| < \alpha$ where $\alpha = 1.5\pi$ is the categorization threshold, otherwise cluster C_2 . Finally, after N trials, we count and compare the number of elements in each group, and choose the average estimation where the corresponding cluster has more elements, as the final estimation for current period.

3.4 Dataset Collection

It is widely known that adequate, curated training data is critical to any Deep Learning approaches. For supervised learning, data is required to have sufficient high-quality labels. Unfortunately, data labeling is typically a manual process done by humans, requires domain knowledge, is slow, arduous, and can be very costly. Furthermore, open datasets of real RF emissions for RX multi-antenna beamforming remain absent. To address this, we devise an efficient multi-stage approach towards building a sufficiently large dataset for training our Deep Learning model, as depicted in Figure 4.

Our setup comprises a single-antenna transmitter (TX) and a multi-antenna receiver (RX). As a first step, we generate random complex samples and save them in the memory of both the TX and RX. Then, the TX transmits the saved samples and the RX collects the samples from receiving branches and saves to files. Because of the channel effects, the received samples will experience unknown phase shifts. To determine these phase shifts, we first chunk the received samples, which we subsequently cross-correlate with the transmitted samples already saved in the RX. The phase shift is calculated as the argument of the peak in the correlation output. The labels (relative phase Δ_θ) are obtained by taking the difference between the two phase shifts. When the channels are static, the acquired labels will have very little variance. This would negatively impact the training and bias the DL model towards a small range of values. To address this, and improve the diversity of the dataset, we employ a simple data augmentation technique. For each chunk of RF samples, we randomly shift the phases by a value in $[-\pi, \pi]$ and adjust the label accordingly. We emphasize that while this process is quite efficient, it is *not necessary* for DEFORM to repeat this process for each type of data (modulation or bandwidth). As we will show in later sections, our deep beamforming system is agnostic to bandwidths and modulations. Thanks to its universality, DEFORM can be quickly deployed without a prior knowledge about the RF signal and channel parameters.

4 EVALUATION

We validate the universality of DEFORM for different modulations and bandwidths under fading wireless channels. To train the DL

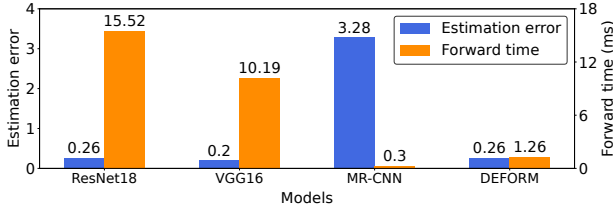


Figure 5: Comparison of the DL models

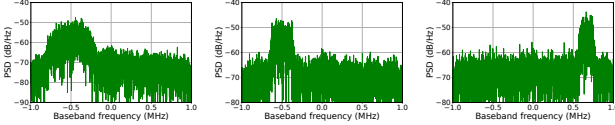
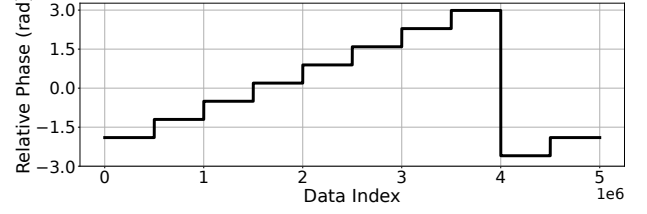


Figure 6: Power Spectral Density plot shows the center frequency offsets in the received wideband signals. It is noted that the y-axis scale is relative.

model, we use the techniques described in Section 3.4 and collect a dataset of over 167 million complex RF samples transformed into 654,553 real-valued tensors of size $2 \times 128 \times 2$, each corresponds to a total of 256 samples collected by the two RX antennas. For this dataset, we use an Ettus USRP B210 software-defined radio (SDR) with the TX implemented using GNURadio [1] to transmit BPSK signals (SNR ranging from 0 to 35 dB) to the RX through two identical coaxial cables with a fixed TX bandwidth of 1 MHz. The RX bandwidth is maintained at 1 MHz for the whole dataset. The dataset is split into training, validation, and test sets with ratio 0.64 : 0.16 : 0.2, respectively. While being trained with a fixed set of RF settings, DEFORM still performs very well on other unseen and more sophisticated settings of modulations, bandwidths, and channels. To the best of our knowledge, our work is the first universal RX beamforming system in the literature that is designed using Convolutional Neural Network (CNN) and extensively evaluated for practical, universal RF beamforming capabilities.

4.1 DL Model Comparison

To validate our design of neural network and highlight the benefits of our model for the specific task of phase estimation, we evaluate and compare the CNN model with three popular CNN architectures: VGG16 [25], ResNet18 [9] and MR-CNN [21] using the test set of our beamforming dataset. All models are implemented using PyTorch library [22], then trained and tested on a NVIDIA GeForce GTX 1080 GPU using Adam Optimizer [14] and ReduceLROnPlateau Learning Rate Decay scheduler [33] with initial learning rate $lr = 0.005$. The evaluation metrics are estimation error and network forward time. Mean Square Error loss function (Equation (10)) is used to calculate the estimation error. We use `torch.cuda.synchronize` from PyTorch to synchronize CUDA operations before and after the network propagation function and calculate the elapsed time of such function accurately. Figure 5 compares the estimation error of models on the test set and the network forward time. It is clear that in terms of estimation correctness, DEFORM outperforms MR-CNN

Figure 7: Emulated fading pattern for over-the-cables evaluation. The initial relative phase is randomly chosen within $[-\pi, \pi]$, then slowly changing throughout the experiment for a total amount of 2π radian.

with a test error of 12.6 times lower (0.26 compared to 3.28). Moreover, DEFORM's CNN is 8 times faster than VGG16 and 12 times faster than VGG16 while having comparable estimation error (equal to ResNet18, and only 0.06 lower compared to VGG16). Compared with these two models, DEFORM is lightweight, more computationally efficient, and can quickly estimate phases with high precision, which makes it more suitable for real-time and embedded systems.

4.2 Over-the-cables Evaluation

We evaluate DEFORM's performance in a relatively idealistic environment where the RF signals propagate through coaxial cables, where multi-path and other fading effects are absent. Data packets are sent from the TX to the two analog inputs of RX (devices are Ettus USRP B210 with SDR) through a pair of identical cables (So the received signals have similar SNRs). TX signal is modulated using differential BPSK, QPSK, 8-PSK, GMSK and 16-QAM techniques, and with a fixed TX bandwidth of 1 MHz and center frequency of 795 MHz. We assess DEFORM's wideband capability by using different values of RX bandwidths, with a random shift of RX center frequency when the bandwidth is larger than 1 MHz (Figure 6).

Evaluation for Model Optimizations. To highlight the impact of the optimization techniques, we measure and analyze the received Bit Error Rate (BER) in five cases: (1) When beamforming is turned off and one of the two received signals is selected for decoding (Note that in our over-the-cables experiments, received signals have similar SNRs), (2) when beamforming is used with the single-output CNN (BF-SO), or (3) with the rotational double-output CNN (BF-DO), and when DEFORM is enabled with double-output CNN and (4) temporal smoothing (DEFORM-TS) or (5) multi-trial averaging (DEFORM-MT) optimization. To calculate BER, we record TX and RX signals at the SDRs and transfer them to the host computer to compare and count the bit errors. Figure 8 illustrates the evaluation results, where we compare the approaches with BPSK transmissions. For each scenario, RX signal features (relative phase, bandwidth) are artificially adjusted to show the effects of model optimizations: For Figure 8a, because Δ_θ is far from the phase boundaries, all beamforming settings including the baseline single-output CNN achieve 3 dB gain compared to non-beamforming. When $\Delta_\phi = \frac{9\pi}{10}$ which is very close to π (where the estimation of the single-output CNN estimating phase in $[-\pi, \pi]$ experiences abrupt variations - Figure 8b), the efficiency of BF-SO decreases to less than 1 dB

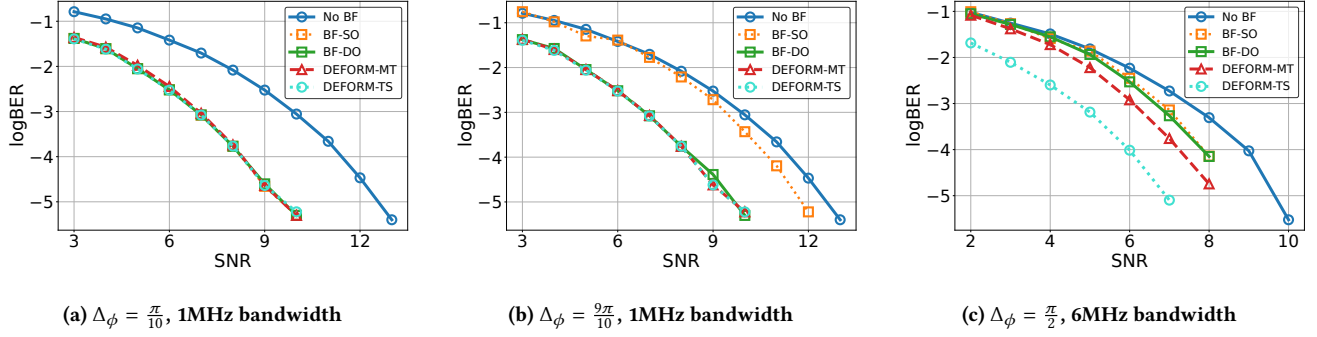


Figure 8: BER comparison between non-beamforming, and beamforming with the baseline CNNs and DEFORM’s optimizations.

gain, while BF-DO and further optimization settings maintain the 3 dB gain. When we increase RX bandwidth to 6 MHz (Figure 8c), DEFORM-TS and DEFORM-MT outperform the baseline CNNs with 3 dB and 2 dB gains compared to non-beamforming, respectively. This justifies the efficiency of DEFORM’s optimizations to address the increasing variations and outliers of the baseline CNNs introduced by communication settings of wider bandwidths.

Evaluation for Signal Features Universality. As mentioned above, we consider various settings of modulations and bandwidths to evaluate the universality of DEFORM. We emulate the slow fading effect of the real wireless channels by artificially adjusting the phases of received signals following a stair-step pattern for the relative phase $\Delta\theta$ as illustrated in Figure 7 to cover the whole 2π phase range. Figure 9 shows the Bit Error Rate (BER) analysis where DEFORM is evaluated with both DEFORM-TS and DEFORM-MT approaches, and compared with non-beamforming. It is evident that for all combinations of modulation and RX bandwidth, DEFORM-TS can provide a 3 dB SNR gain compared to using a single RX branch, with the only exception of GMSK-2MHz where the gain for $\text{BER} = 10^{-5}$ is approximately 2.5 dB. Meanwhile, DEFORM-MT also achieves 3 dB gain for 1 and 2 MHz RX bandwidths, and about 2-3 dB gain for 4 MHz bandwidth. However, with 6 MHz bandwidth, the SNR gain of DEFORM-MT declines to 1 dB for BPSK and QPSK at $\text{BER} \geq 10^{-4}$, and to 2 dB for 8-PSK and 16-QAM. Interestingly, it still performs equally well with DEFORM-TS for GMSK in 6 MHz RX bandwidth, both having a 3 dB gain for all BER levels.

5 DISCUSSION AND RELATED WORK

It is clearly seen that the Deep Learning-based approach of DEFORM can be extended to larger multi-antenna systems to achieve even higher than 3 dB gain. For a RF receiver system of N receiving antenna elements, the deep learning architecture can be modified to have $2 \times (N - 1)$ outputs in which two estimations are made for each relative phase between the pre-selected antenna and one other antenna. The new requirement of such systems are time synchronization mechanisms for RX radios as typical wireless peripherals have a limited number of antennas (Most of Ettus’s USRPs only support 2 simultaneous RX channels [24]).

Our current system assumes that there is only one user in the observing spectrum at any given time. The problem becomes more challenging when multiple users are simultaneously transmitting

on the same frequency band, which results in significant interference that damages the phase structure in the captured RF samples. In this case, the prediction could be accurate for only one user, or even not working for any users. It would be interesting to investigate the CNN’s capability for multi-band spectrum and collision analysis in the future work.

Theoretical aspects of beamforming have been investigated in the literature [4, 19, 28, 30, 34], including some efforts to build the so-called *blind beamformers* [4, 10, 34], that estimates the channel phase offsets without explicit knowledge from the transmitter. The practicality of those methods and systematic deployment guidelines, however, are still open questions. Popular wireless communications technologies still utilize *informed beamformer* approaches that estimate the channel using information from the transmitter such as pilot sequence [35] in IEEE 802.11 or reference signal [17] in 5G radios. As discussed, this approach typically requires significant communication overhead, and is limited to specific TX-RX setup and communication techniques.

Advances in Machine Learning and Deep Learning have emerged as a solution for critical problems in various areas, including wireless communications. In recent years, ML and DL approaches have been extensively utilized in various tasks such as modulation recognition [21], RF technology identification [20], or wireless localization [29]. There are also some efforts to enable ML-driven beamforming. For example, in [32], a Deep Neural Network (DNN) is developed for OFDM channel estimation. In [11], DNN is also utilized for downlink MIMO beamforming. Nonetheless, those works are limited by simulated data, and specific assumptions of channel model. Furthermore, they lack the explanation and evaluation for the impact of various RF characteristics, i.e. link modulations and bandwidths. Compared to those, our work is unique in two main aspects. First, our CNN-based DL model is trained with real RF data acquired by an efficient dataset collection process. Second, our system is agnostic to different RF settings of modulations and bandwidths. Despite being trained on a fixed, basic RF settings, we can still achieve the *optimal* beamforming gain in complex, unseen settings. Hence, DEFORM can be used as a universal RX beamforming module for existing multi-antenna RF receivers. As the next step, we will apply DEFORM to RF receiver operating in realistic over-the-air environments with various wireless channel artifacts (e.g., multi-path fading). We will also justify DEFORM’s universality

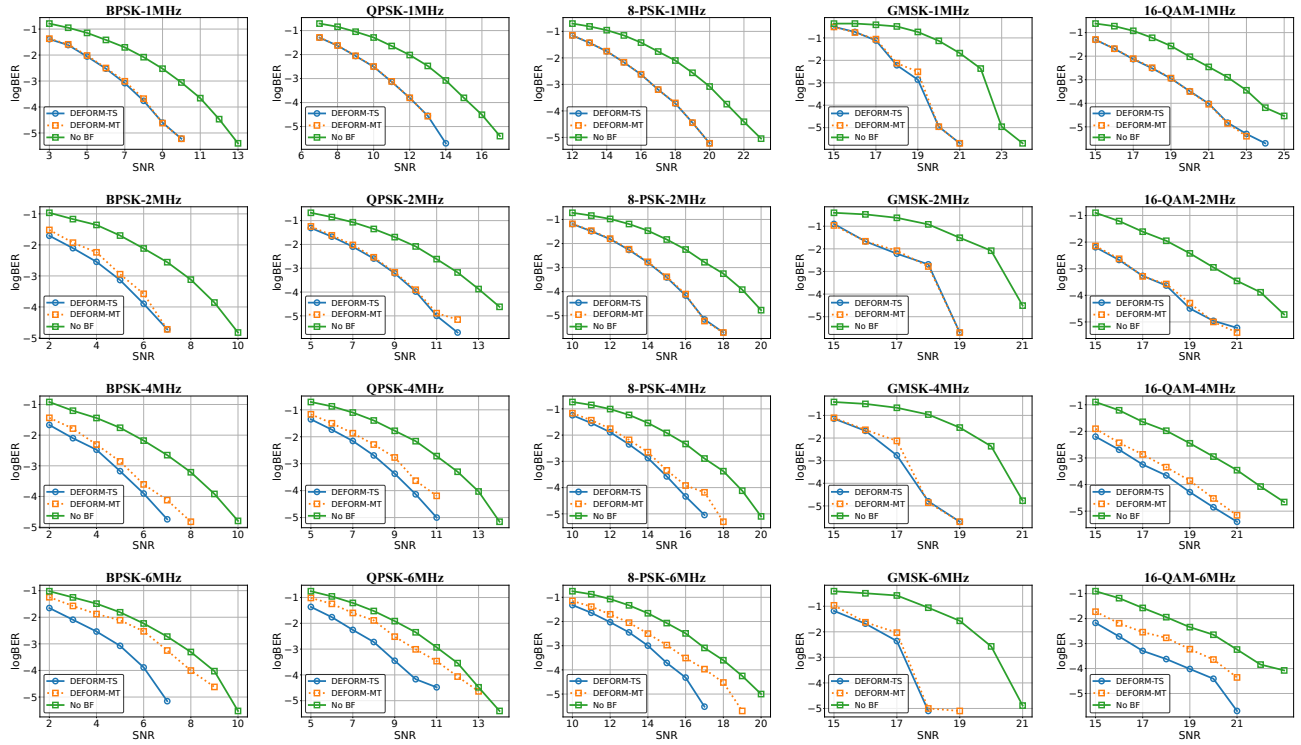


Figure 9: BER analysis of over-the-cables communications for different modulation schemes and RX bandwidths. DEFORM is evaluated with Temporal Smoothing (DEFORM-TS) and Multi-Trial Averaging (DEFORM-MT).

with the beamforming-relay application supporting the communications of different wireless technologies such as ZigBee and LoRa.

ACKNOWLEDGMENTS

This work was partially supported by grants NAVY/N00014-20-1-2124, NCAE-Cyber Research Program, and NSF/DGE-1661532.

REFERENCES

- [1] 2021. GNURadio. <https://www.gnuradio.org>
- [2] O. Abdel-Hamid et al. 2014. Convolutional Neural Networks for Speech Recognition. *IEEE/ACM TASLP* (2014).
- [3] C. M. Bishop. 2006. *Pattern Recognition and Machine Learning*.
- [4] J.F. Cardoso et al. 1993. Blind beamforming for non-gaussian signals. *IEEE Proceedings F* (1993).
- [5] CISCO Meraki. 2018. SNR and Wireless Signal Strength .
- [6] X. Glorot et al. 2011. Deep Sparse Rectifier Neural Networks. In *AISTATS'11*.
- [7] Andrea Goldsmith. 2005. *Wireless Communications*. Cambridge University Press.
- [8] I. Goodfellow et al. 2016. *Deep Learning*.
- [9] K. He et al. 2016. Deep Residual Learning Image Recognition. In *CVPR'16*.
- [10] I. Himawan et al. 2011. Clustered Blind Beamforming From Ad-Hoc Microphone Arrays. *IEEE TASLP* (2011).
- [11] H. Huang et al. 2018. Unsupervised learning-based fast beamforming design for downlink MIMO. *IEEE Access* (2018).
- [12] S. Ioffe et al. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML'15*.
- [13] N. Kalchbrenner et al. 2014. A Convolutional Neural Network for Modelling Sentences. *arXiv:1404.2188*
- [14] D. Kingma et al. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*
- [15] S. Kumar et al. 2014. Accurate indoor localization with zero start-up cost. In *ACM Mobicom'14*.
- [16] H.J. Kwon et al. 2019. Machine Learning-Based Beamforming in Two-User MISO Interference Channels. In *ICAIC'19*.
- [17] X. Lin et al. 2019. 5G New Radio: Unveiling the Essentials of the Next Generation Wireless Access Technology. *IEEE CSM* (2019).
- [18] W. Liu et al. 2010. *Wideband Beamforming: Concepts and Techniques*.
- [19] C. Masouros et al. 2015. Exploiting Known Interference as Green Signal Power for Downlink Beamforming Optimization. *IEEE TSP* (2015).
- [20] H.N. Nguyen et al. 2021. Wideband, Real-time Spectro-Temporal RF Identification. In *ACM MobiWac'21*.
- [21] T. O'Shea. 2016. Convolutional Radio Modulation Recognition Networks. In *Engineering Applications of Neural Networks*.
- [22] A. Paszke et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NIPS'19*.
- [23] D. Rahamim et al. 2004. Source localization using vector sensor array in a multipath environment. *IEEE TSP* (2004).
- [24] Ettus Research. 2021. Products. <https://www.ettus.com/products/>
- [25] K. Simonyan and A. Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. In *arXiv:1409.1556*.
- [26] G. Stuber et al. 2004. Broadband MIMO-OFDM wireless communications. *Proc. IEEE* (2004).
- [27] H.L. Van Trees. 2004. *Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory*. Wiley.
- [28] V. Venkateswaran et al. 2010. Analog Beamforming in MIMO Communications With Phase Shift Networks and Online Channel Estimation. *IEEE TSP* (2010).
- [29] X. Wang et al. 2016. CSI-based fingerprinting for indoor localization: A deep learning approach. *IEEE TVT* (2016).
- [30] Q. Wu et al. 2019. Intelligent Reflecting Surface Enhanced Wireless Network via Joint Active and Passive Beamforming. *IEEE TWC* (2019).
- [31] J. Xiong et al. 2013. ArrayTrack: A Fine-Grained Indoor Location System. In *NSDI'13*.
- [32] H. Ye. 2018. Power of Deep Learning for Channel Estimation and Signal Detection in OFDM Systems. *IEEE WCL* (2018).
- [33] M. Zaheer. 2018. Adaptive Methods for Nonconvex Optimization. In *NIPS'18*.
- [34] L. Zhang et al. 2017. An Eigendecomposition-Based Approach to Blind Beamforming in a Multipath Environment. *IEEE CL* (2017).
- [35] Z. Zhao et al. 2013. Channel Estimation Schemes for IEEE 802.11p Standard. *IEEE ITSM* (2013).