Deep Learning-based Carrier Frequency Offset Estimation with One-Bit ADCs

Ryan M. Dreifuerst, Robert W. Heath Jr.
Department of Electrical and Computer Engineering
The University of Texas at Austin
Email: {ryandry1st, rheath}@utexas.edu

Mandar N. Kulkarni, Jianzhong Charlie Zhang
Standards and Mobility Innovation Lab
Samsung Research America
Email: {mandar.kulkarni, jianzhong.z}@samsung.com

***Abstract*—Low resolution architectures are a power efficient solution for high bandwidth communication at millimeter wave and terahertz frequencies. In such systems, carrier synchronization is important yet has not received much attention. In this paper, we develop and analyze deep learning architectures for estimating the carrier frequency of a complex sinusoid in noise from the 1-bit samples of the in-phase and quadrature components. Carrier frequency offset estimation from a sinusoid is used in GSM and is a first step towards developing a more comprehensive solution with other kinds of signals. We train four different deep learning architectures each on eight datasets which represent possible training considerations. Specifically, we consider how training with various signal to noise ratios (SNR), quantization, and sequence lengths affects estimation error. Further, we analyze each architecture in terms of scalability for MIMO receivers. In simulations, we compare computational complexity, scalability, and mean squared error (MSE) versus classic signal processing techniques. We demonstrate that training with quantized data, drawn from signals with SNRs between 0-10dB tends to improve deep learning estimator performance across the entire SNR range of interest. We conclude that convolutional models have the best performance, while also scaling for massive MIMO situations more efficiently than FFT methods. Our approach is able to accurately estimate carrier frequencies from 1-bit quantized data with fewer pilots and lower signal to noise ratios (SNRs) than traditional signal processing methods.**

***Index Terms*—Carrier frequency offset, millimeter wave, MIMO, deep learning, one-bit receivers.**

I. Introduction

Low resolution data converters reduce power consumption in wideband millimeter wave (mmWave) and terahertz systems. Many aspects of communication have been explored for systems with low resolution analog to digital converters (ADCs) including equalization [1] and channel estimation [2]. Unfortunately, synchronization in general, and carrier frequency offset estimation specifically, have not received much attention. Synchronization, though, is a critical first step in wireless cellular.

The challenges of low resolution are amplified at higher carrier frequencies, as they are normally accompanied by higher bandwidth channels. Yet the benefits of low resolution are also greater at these frequencies, especially when considering the important role of multiple antennas to overcome low SNR with array gain. Low resolution further enables fully digital architectures that can leverage more sophisticated types of MIMO operations as well, including multi-stream and multi-user. As with any MIMO communication system [3], training signals are both a means for synchronization and channel estimation, but also a source of overhead. This motivates developing algorithms that work with small training sequences to reduce the overhead penalty.

In this paper, we develop approaches for frequency estimation based on machine learning. We set up the one-bit frequency estimation problem as a learning problem, where we estimate the nonlinear function from quantized in-phase and quadrature samples. We divide the learning problem into two categories, neural network (NN) architecture selection and training methodology. Based on our initial experiments, we decided to consider four different networks: a feedforward NN, a convolutional NN, a recurrent NN, and a variation of convolutional neural networks called residual neural networks. We trained each network using simulated data as shown in Table I. The use of simulated data is based on convenience; we would expect deep learning would work even better for more complicated signal models. We compare machine learning methods to three traditional estimators: the classic periodogram, Welch's periodogram [4], and the MUSIC estimator [5]. These estimators are well suited to high resolution sequences of data with reasonably high SNR, but fail under conditions expected for next generation wireless systems. We show that our estimators can outperform these benchmarks by up to 15dB with one-bit resolution. Finally, we test the computational complexity for our estimators by comparing the execution time versus block length and number of RF chains in MIMO systems. Trained neural networks easily scale for massive MIMO systems by performing efficient matrix multiplication in graphics processing units (GPU). From these simulations, we conclude deep learning is a reasonable approach for frequency estimation for future wireless communication systems.

II. Models for frequency estimation

In this section, we introduce the frequency estimation problem. We start by reviewing three classic approaches assuming full-resolution. We continue by reviewing relevant work designed for one-bit resolution. Then, we conclude by presenting the specific quantized system model that will be used for learning.

A. Conventional approaches to sinusoid estimation in noise

We consider the classic problem of estimating the frequency of a discrete-time set of samples, $\{z[n]\}_{n=0}^{N-1}$, resulting from sampling at the receiver in noise. Let A denote the amplitude, θ the phase shift, ω_0 the frequency to be estimated, and $v[n]$ as unit variance complex AWGN. The received signal model is

$$z[n] = Ae^{\mathrm{j}(\omega_0\, t_n+\theta)} + v[n]. \tag{1}$$

We benchmark against three estimators based on (1), two non-parametric FFT methods and one parametric method.

The well known maximum-likelihood estimator based on the system model (1), is the peak of the periodogram. Nevertheless, the original periodogram is not a consistent estimator. A variant of the original periodogram, Welch's method, reduces the noise in the frequency estimation at the cost of resolution and is a consistent estimator [4]. The latter method works well for large sample sizes but fails for short sequences. An alternative estimator which is robust to short sequences is the multiple signal classification estimator [5]. MUSIC uses eigendecomposition of the covariance matrix along with knowledge of the number of frequency components and sample points. It is especially effective for small data lengths but is more computationally complex than FFT approaches. We use the two periodogram methods and MUSIC as benchmarks for comparison.

B. Relevant work for estimation with one-bit resolution

Prior approaches to one-bit resolution techniques begin by using a quantized system model that follows from (1). In this model, $\text{sign}(\cdot)$ is the nonlinear one-bit quantization function which maps non-negative values to 1 and negative values to -1. This is applied separately to the in-phase and quadrature components, by applying $\text{sign}(\cdot)$ to the real, $\mathbb{R}\text{e}(\cdot)$, and imaginary, $\mathbb{I}\text{m}(\cdot)$, parts separately, as shown in (2)

$$x[n] = \text{sign}(\mathbb{R}\text{e}[z[n]]) + \mathrm{j}\,\text{sign}(\mathbb{I}\text{m}[z[n]]). \tag{2}$$

The one-bit frequency estimation problem is to then find an estimate of ω_0 from $\{x[n]\}_{n=0}^{N-1}$.

Some research has gone into CFO estimation with one-bit resolution [6]–[9]. In [6] the effects of quantization are considered and an asymptotic Cramér-Rao bound is derived along with fast correlation based estimators. The correlation algorithm, while efficient, does not consider how poor SNR affects the estimator. In [7] dithering is used to improve results for one-bit estimation, but ultimately FFT based methods are found to be superior for estimation with sinusoids. Time-varying thresholds are used in [8]. It requires, however, a course and fine grid search method, which is computationally inefficient, used many samples, and required knowledge of the signal amplitude. More recently, an algorithm using convex lifting and EM-GAMP was proposed in [9], but noted that the use of convex lifting is not computationally efficient. While many algorithms have been introduced which solve the one-bit frequency estimation problem, the issue of carrier synchronization is still yet to be solved for envisioned applications [10].

TABLE I
TABLE SHOWING DATASET STYLES FOR EACH OF THE SETS USED FOR NEURAL NETWORK TRAINING AND TESTING.

Dataset	SNR Range [dB]	Resolution	Sequence Length, N
1	0 to 10	1-bit	8
2	0 to 10	1-bit	16
3	0 to 10	1-bit	32
4	0 to 10	Full	8
5	0 to 10	Full	16
6	0 to 10	Full	32
7	10 to 20	Full	16
8	10 to 20	1-bit	16
Test Set 1	-20 to 40	1-bit	8
Test Set 2	-20 to 40	1-bit	16
Test Set 3	-20 to 40	1-bit	32

C. Going towards a learning solution

Now we reformulate the frequency estimation problem slightly to make it more amenable to conventional deep learning methods. As most network architectures have not been extended to the complex case, we separate the real and imaginary components of $z[n]$ in (1) into two real values which are stacked into a column vector $\boldsymbol{x}[k]$ of the form shown in (3)

$$\boldsymbol{x}[n] = [\text{sign}(\mathbb{R}\text{e}[z[n]]),\ \text{sign}(\mathbb{I}\text{m}[z[n]])]^T. \tag{3}$$

The in-phase and quadrature components are now used as two separate features. The objective of this paper is to learn a function, $\phi(\cdot)$, that takes $\{\boldsymbol{x}[0], \boldsymbol{x}[1], ..\, \boldsymbol{x}[N-1]\}$ and produces an estimate of the carrier frequency, $\hat{\omega}_0$, without knowledge of the SNR or phase offset, resulting in the learning model

$$\hat{\omega}_0 = \phi(\boldsymbol{x}[0], \boldsymbol{x}[1] ... \boldsymbol{x}[N-1]). \tag{4}$$

We now present an approach to finding the function $\phi(\cdot)$ from (4) through the use of deep learning.

III. DEEP LEARNING APPROACH

We develop a deep learning approach to carrier frequency offset estimation, attempting to find the best estimate of the nonlinear function from (4). We divide the task into two areas: training methodology and architecture selection. We present important dataset considerations for training a model and four architecture styles to evaluate.

A. Dataset generation

We configure the training data using three different qualities: the number of pilot symbols, the SNR, and quantized/unquantized training data. The naive approach would be to make the data as similar to the test case as possible, specifically, quantizing the data and making the SNR range across the spectrum of interest. When the SNR is below 0dB, however, the sampled data is badly distorted by noise and so a model trained on this set would not learn the correct features for accurate frequency estimation. We show how the selection of training data affects the model mean squared error in Section IV. Training is done by generating the data with the desired parameters and uniformly distributed carrier frequencies, dividing it into a training and validation set, and iteratively updating the parameters until the MSE evaluated on the validation set stops improving with learning rate decay. The phase offset, θ, is assumed unknown but drawn from a uniform distribution, $\theta \in [0, 2\pi]$. The dataset selections are summarized in Table I.

1) Training symbols: We develop models based on three different block lengths: 8, 16, and 32 symbols. In each case, the received signal is sampled and quantized from a sinusoid with a carrier frequency we wish to estimate. Each model is trained with each set length for comparable performance evaluation.

2) Training SNR: We propose two different ranges of SNR for training the models, while still evaluating the algorithm across the range from -20dB to 40dB. The two considered ranges are low and high SNR, where low SNR includes symbols with signal to noise ratios between 0dB and 10dB, and high SNR ranges from 10db to 20db. Extending training outside of these spans does not allow the model to either: learn the data if there is more noise, or learn to generalize if there is almost no noise.

3) Quantization: The quantization function is critical for the algorithm to learn, as it is highly nonlinear and largely the reason the task is difficult, due to the destruction of information. We evaluate two cases, one in which the model is trained with purely one-bit quantized data, and another where the model is given unquantized data. In each case, the models are still evaluated using only the quantized data.

B. Model architectures

We begin with the classic building blocks of neural network (NN) layers. Specifically, we wish to understand how feed-forward, convolutional, and recurrent networks impact performance in estimating sinusoid frequency, given the learning model, (4). In Section IV we further demonstrate how these models are impacted by the various dataset considerations from Section III-A and how the computational complexity scales. In each case, unless otherwise specified, the activation function rectified linear unit (ReLU) is used, weights are updated with Adam optimizer and learning rate decay, and the final layer is a regression layer with a single output, which is the estimate of the digital frequency, $\hat{\omega}_0$. Finally, we investigate a modern architecture style known as a residual layer [11], which uses a bypass method and summation to expand the solution space of a layer. We present results for one network architecture in each case, selected based on experiments with different hyperparameters to find the one that is most effective across the datasets. A Jupyter notebook containing all relevant code is available at [12].

1) Artificial neural network: An artificial neural network (ANN) is the simplest variety, modeled after the brain. In this setup, all of the nodes are fully connected, so that the input to each hidden layer is an affine transformation of the previous layer's output after the activation function. The weights and biases are learned through gradient descent. We expect for an ANN with sufficient model complexity and data, the function should be learnable [13], even if the performance is highly dependent on the SNR. The model we use is made of 1024 nodes followed by five hidden layers of 512, 256, 128, 64, and 32 nodes.

2) Convolutional neural network: A convolutional neural network (CNN) provides a convolution feature by applying a filter map and convolving it over the input data. This is very reminiscent of many tools in signal analysis which rely on convolution for time domain data. The connection between classical signal processing and the convolutional network lends credibility to the use of CNNs for frequency estimation. CNNs are also known to be exceptional at extracting fundamental features from local repetitive fields [14], which occur regularly in quantized sinusoids. Unless otherwise specified, the filter is a 1D convolution, with a filter size of three, stride length of one, and zero padding. The convolutional model we investigate is comprised of six convolutional layers starting with 8 filters and doubling each time, followed by two fully connected layers each with 128 and 64 units.

3) Recurrent neural network: A recurrent neural network (RNN) improves on the simple weights and bias terms of an ANN by adding memory to the node. The exact implementation depends on if it is a gated recurrent unit (GRU) or a long-short term memory (LSTM) layer which adds a forget gate, however, both make use of memory in resolving the output. We use the standard LSTM implementation, which uses hyperbolic tangent activation and sigmoid activation for the recurrent components. The standard model tested has three LSTM layers starting with 16 units and doubling each time, followed by 128 and 64 fully connected units, similar to the CNN model.

4) Residual layer: Residual networks were proposed for the Imagenet competition in 2015 [11]. In this architecture, the inputs are transformed through a layer, then added back with the original input and sent to the next layer. The insight provided by [11] relies on a simplification of learning, where the first term in the series is the bypassed input and then only the remaining component, the residue, must be learned as a nonlinear function, similar to a Taylor series expansion. This method is applied to convolutional layers to form residual blocks. The residual model we test is the same as the convolutional one, but after the first layer, all convolution layers are replaced by residual blocks.

IV. SIMULATION

Models are compared by evaluating the mean squared error over signal to noise ratios from -20dB to 40dB and digital frequency estimation for $\omega_0 \in [0.2, 0.25]$ with a uniform distribution and taken over 1000 Monte Carlo runs per half dB SNR. We first show that deep learning performs as well or better than the benchmark methods for short, quantized sequences in Fig. 1. For the benchmark algorithms, we use zero padding, parabolic interpolation, and, in the case of

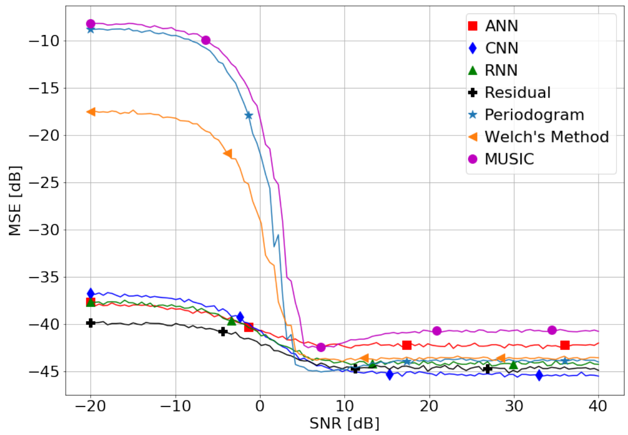

Fig. 1. This figure shows the MSE for the four neural network estimators along with the three benchmark estimators, the original periodogram, Welch's method, and MUSIC, in estimating the carrier frequency. The simulation is performed with N=16 symbols, trained on quantized, low SNR data. Deep learning estimators can provide improvements for short sequences, especially for data around -5dB, where the improvement is on the order of 15dB over classic estimators. For high SNR, all of the estimators except the correlation based estimator, MUSIC, approach similar estimation error near -45dB.

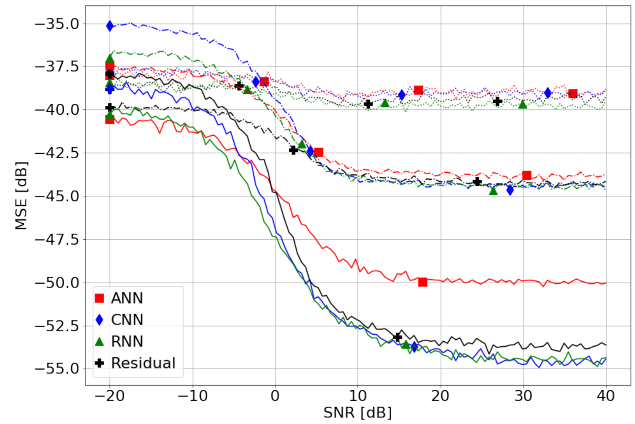

Fig. 2. A comparison of neural network architectures MSE performance for pilot lengths N=8 $(\cdot\cdot\cdot)$, N=16 $(-\cdot)$, and N=32 $(-)$ symbols. All networks were trained with unquantized, low SNR data. We see both convolutional and memory methods have better estimation error than the simpler ANN for the N=32 sequences, suggesting that there is some information in the data which is not learned by the ANN.

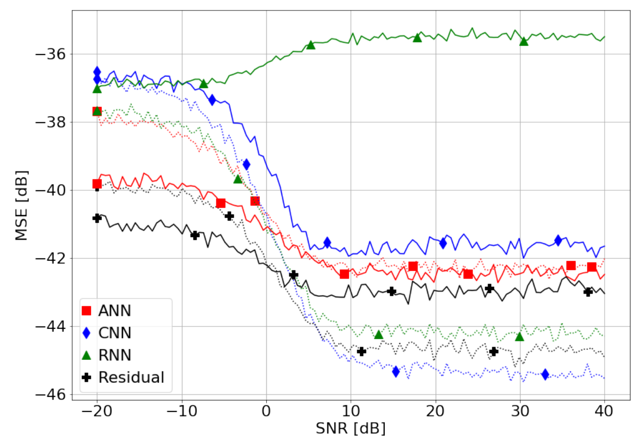

Fig. 3. This figure shows a plot of the MSE for neural network estimators trained with quantized, low $(\cdots)$ and high $(-)$ SNR data. Networks are trained with 16 symbol sequences and signal to noise ratios in the range of 0-10dB for the low training method and 10-20dB for the high training method. The use of low SNR training data generally improves model performance across the entire range of SNR testing.

Welch's method, set overlap factor to 50% [4]. All four models are better than the benchmarks when the SNR is below 3dB, and asymptotically approach similar error for high SNR. Next, we investigate how varying these parameters changes the performance.

1) Block length: As is expected with longer pilot lengths, the models tend to do better, mimicking other estimators. The MSE performance of each structure is shown in Fig. 2. We can see for sequences with only 8 symbols, the neural networks tend to have estimation error between -37.5dB and -40dB throughout the range of interest, which is only marginally better than simply estimating the mean of ω_0, near -37dB. In the case of 16 or 32 symbols, we note that the performance is distinctly better than a first order estimate of the frequency. Importantly, this means the model is likely biased to the training frequency range, however, this is acceptable in many applications where the frequency is standardized such that the carrier frequency will not vary outside the range of training. Investigation into this bias and generalizability to wider frequency ranges is left to future work.

2) Training SNR: We show that the use of high SNR data, signals in the 10dB - 20dB range, leads to universally worse estimators for test SNR above 0dB in Fig. 3. The training SNR is especially critical to the recurrent neural network, which is unable to learn the function and diverges as SNR increases, which is counter to the naive expectation that the model would learn very well when training data is similar to testing data.

3) Quantization: Applying one-bit quantization to the training data produces models which should have very similar data between training and testing. On the one hand, the data similarity should produce improvements. On the other hand, quantized data has very limited information which may make the frequency estimation function unlearnable, especially for short pilot sequences. We show that the effects of quantization are advantageous for more complex models (CNN, RNN, residual), but impede the learning process for simple models (ANN) in Fig. 4.

A. Computational complexity

Neural networks can be notoriously computationally complex, which is one reason they are often ignored in near real-time applications. In the last few years, extensive research has gone into improving the efficiency and inference time for these models. We do not consider any extraneous methods, relying simply on using a Tesla P100 GPU and standard implementations with Tensorflow [15]. We test both the scalability for models computed with N=8, 16, and 32 pilot symbols, as well as the parallelizability for K = 1, 10, 100, and 1000 received

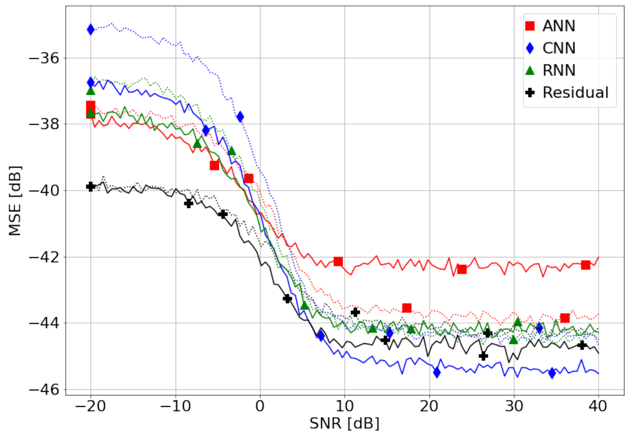

Fig. 4. The results of evaluating networks trained with different resolution data. Networks are trained using N = 16 pilot symbols and low SNR. The one-bit quantization trained models (—) are compared with the unquantized training models (···). Except for the ANN, the models tend to improve with the use of quantization on the training data.

TABLE II
EXECUTION TIME FOR CLASSICAL AND DEEP LEARNING ESTIMATORS WITH BLOCK LENGTHS OF N={8, 16, 32, 128}

Algorithm	N=8 [ms]	N=16 [ms]	N=32 [ms]	N=128 [ms]
FFT	28.9	29.1	29.1	29.4
Welch's	34.8	34.9	34.9	34.9
ANN	26.3	26.2	26.1	26.9
CNN	28.5	28.8	29.6	32.2
RNN	62.9	93.3	154	515
Residual	27.9	27.5	27.6	27.9

TABLE III
EXECUTION TIME VERSUS RF CHAINS FOR N=32 PILOT SYMBOLS

Algorithm	K=1 [ms]	K=10 [ms]	K=100 [ms]	K=1000 [ms]
FFT	0.359	3.07	29.1	295
Welch's	0.427	3.38	31.6	401
ANN	23.7	23.9	26.5	53.3
CNN	24.6	24.4	29.1	67.7
RNN	60	57.4	156	1060
Residual	24.6	24.2	27.9	55.5

sequences, representing the receiver data from K different RF chains in a massive MIMO system. These results are shown in Tables II and III. Most importantly, we see all models except for the RNN scale well versus the block length and number of executions. The three neural networks have shorter computational times than the benchmark algorithms for 100 or more signals for any length of sequences. This is largely due to well designed GPU implementations of neural networks allowing for efficient matrix-matrix multiplication. We test the execution time of the FFT and Welch's method with zero padded sequences of 2^{12} samples.

V. CONCLUSION

We demonstrate the effectiveness of deep learning for one-bit frequency estimation of a single sinusoid in noise. Current methods are too computationally intensive, sensitive to noise, or fail to scale for MIMO antenna systems. We show that the use of low SNR, quantized training data with short pilot sequences leads to estimators with better mean squared error than FFT methods. We also show how the neural network computation time scales more efficiently for massive MIMO situations than traditional signal processing techniques. Finally, we conclude by suggesting that convolutional networks and their variants are well situated to be robust and efficient for complex signal processing. In future work, we will expand our investigation to include phase and channel estimation, while also considering the use of special waveforms in synchronization with one-bit quantization. Finally, we will investigate how our algorithms perform on experimental measurements.

ACKNOWLEDGMENT

The authors would like to thank Samsung Research America for supporting this work.

REFERENCES

[1] K. Kotera, O. Muta, and H. Furukawa, "Efficient nonlinear equalization scheme for MIMO constant envelope modulation receivers affected by quantization error," in *Proc. of ICOIN*, 2012, pp. 275–279.
[2] H. Xie, F. Gao, and S. Jin, "An Overview of Low-Rank Channel Estimation for Massive MIMO Systems," *IEEE Access*, vol. 4, pp. 7313–7321, 2016.
[3] R. W. Heath Jr. and A. Lozano, *Foundations of MIMO Communication*. Cambridge University Press, 2018.
[4] P. D. Welch, "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms," *IEEE Trans. Audio Electroacoust.*, vol. 15, no. 2, pp. 70–73, 1967.
[5] R. O. Schmidt, "Multiple Emitter Location and Signal Parameter Estimation." *IEEE Trans. Antennas Propagat.*, vol. 34, no. 3, pp. 276–280, March 1986.
[6] A. Høst-Madsen and P. Händel, "Effects of sampling and quantization on single-tone frequency estimation," *IEEE Trans. Signal Processing*, vol. 48, no. 3, pp. 650–662, 2000.
[7] O. Dabeer and A. Karnik, "Signal parameter estimation using 1-bit dithered quantization," *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5389–5405, Dec. 2006.
[8] C. Gianelli, L. Xu, J. Li, and P. Stoica, "One-bit compressive sampling with time-varying thresholds: Maximum likelihood and the Cramér-Rao bound," in *Proc. of Asilomar Conf. on Signals, Systems and Computers*, 2017, pp. 399–403.
[9] N. J. Myers and R. W. Heath, "Joint CFO and channel estimation in millimeter wave systems with one-bit ADCs," in *Proc. of CAMSAP*, Dec. 2017, pp. 1–5.
[10] J. Liu, Z. Luo, and X. Xiong, "Low-Resolution ADCs for Wireless Communication: A Comprehensive Survey," *IEEE Access*, vol. 7, pp. 91 291–91 324, 2019.
[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of CVPR*, Dec. 2016, pp. 770–778.
[12] R. M. Dreifuerst, "Deep learning for frequency estimation." [Online]. Available: https://github.com/Ryandry1st/Carrier-Frequency-Offset
[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.
[14] Y. LeCun and Y. Bengio, *Convolutional Networks for Images, Speech, and Time Series*. Cambridge, MA, USA: MIT Press, 1998, p. 255–258.
[15] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/