Auto-Regressive RF Synchronization Using Deep-Learning

Michael Petry*,†, *Student Member, IEEE,* Benjamin Parlier*,‡, Andreas Koch*,†, Martin Werner†
*Airbus Defence and Space GmbH, †Technical University of Munich, ‡RWTH Aachen

***Abstract*—This work presents a novel pilot-less Deep-Learning-based synchronization mechanism that seamlessly integrates within state-of-the-art auto-encoder-based end-to-end communication systems. By re-using the idea of Radio Transformer Networks, an auto-regressive strategy is designed that learns to estimate and mitigate synchronization-related perturbations for arbitrarily modulated continuous communication, i.e., sample time offset (STO) and carrier frequency offset (CFO). A performance gain of 0.6 dB in the high-SNR regime compared to classic synchronization techniques is demonstrated. The strength of this approach is a shift from sample-by-sample to batch-wise processing according to the ML paradigm, which enables efficient and fast computation required for practical deployment scenarios using hardware-accelerated ML inference engines.**

***Index Terms*—RF synchronization, algorithm, auto-regressive, machine learning, sample time offset, center frequency offset, RF front end**

I. Introduction

"Let's assume the system is synchronized" is a phrase one cannot avoid when educating oneself on signal processing algorithms within the domain of wireless communication [1]. Whether it's the subject of channel equalization, error correcting codes, signal detection, or decryption, these and much more take a synchronized system for granted. This becomes clear when trying to operate even the simplest digital communication system on real hardware. Without synchronization nothing works in practice, while everything works flawlessly in theory. Unarguably, synchronization is an indispensable component of every digital communication system, and with the introduction of Artificial Intelligence (AI) techniques into next-generation communications systems, the issue of synchronization demands to be re-explored.

Seven years ago, the hype of Machine Learning (ML) sparked over to the RF domain [2] and resulted in a staggering echo within the community. This resonance has manifested itself in several ways, such as the creation of a dedicated Emerging Technology Initiative at IEEE ComSoc [3], the introduction of various AI/ML study items at 3GPP for 6G technologies [4], and in the creation of dedicated ML-tailored venues[1]. Related research is spanning across the RF domain, ranging from neural equalization [5] and graph NN-based channel decoding [6] in SISO systems to advances in cell-free Massive MIMO [7], capacity-orchestration and -scheduling techniques [8], data-aware re-transmission [9], and much more.

Although most research is simulation-based, recent over-the-air communication experiments were inevitably exposed to the issue of synchronization, which created awareness of the algorithmic gap of ML-based synchronization strategies.

In 2016, O'Shea *et al.* performed the first attempt of explicit ML-based synchronization on recorded IQ-samples by leveraging learned attention models and appropriate domain transformations, framed in the so-called "Radio Transformer Network" (RTN) [10]. However, this strategy was not targeted for communication but for signal classification. In 2017, Dörner *et al.* trained an external frame-synchronization module based on sequence decoding to enable continuous over-the-air transmission [11]. However, their setup is based on block-wise auto-encoders, which is known to suffer of performance degradation compared to its bit-wise counterpart, introduced 3 years later [12]. Other attempts exploit either preambles [13] or carrier aggregation [12] in the OFDM modulation scheme. Lastly, special circumstances, such as burst-wise very short packet communication, allows successful reception despite intentionally neglecting synchronisation mechanisms [14].

Up to date, there is a lack of ML-based strategies that solve this issue *explicitly* and therefore a pillar that enables success in conventional systems is missing. With this work we want to give an initial impetus to strengthening this foundation.

The main contribution of this work is the proposition of a novel ML-based auto-regressive synchronization strategy that operates on the physical layer, is pilot-less, supports arbitrarily-shaped modulation schemes, and integrates natively into state-of-the-art AI-based communication systems.

This paper is structured as followed: We revisit the problem of synchronization in Sec. II by reviewing the sources of signal perturbation and their impact on signal integrity, and cover popular (classic) mitigation strategies. We present our novel AI-based synchronization mechanism as well as its training strategy in detail in Sec. III. A first evaluation on its performance, robustness, and competitiveness to classic counterparts is provided in Sec. IV. Sec. V concludes this work and hints possible algorithm improvements. This is followed by an outlook on remaining challenges for making ML-based synchronization practically useful.

This work is supported by the German Aerospace Center (DLR) under project Machine Learning on Telecommunications Satellites (MaLeTeSa) with Grant number 50 YB 2103. Corresponding author: Michael Petry (e-mail: michael.petry@airbus.com).

[1]For a non-exhaustive list please see: https://mlc.committees.comsoc.org/workshops-tutorials-symposia/.

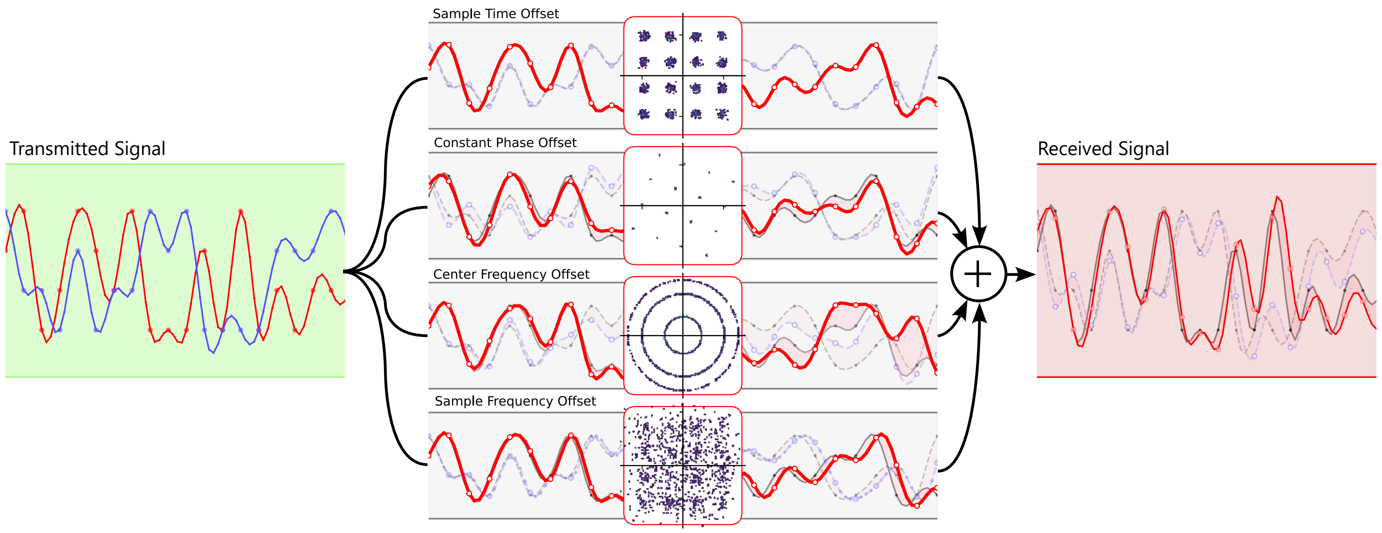

Fig. 1. Visualization of synchronization requirements in the baseband for a 16-QAM modulated signal. Left: Transmitted signal. Right: Received Signal. Red and blue colored lines denote the in-phase and quadrature components of the baseband signal, respectively. Center: Sample Time Offset, Constant Phase Offset, Center Frequency Offset, and Sample Frequency Offset individually illustrated on the full signal. The insets visualize individual impacts on the constellation diagram.

II. The Synchronization Problem - Source, Impact, and Classic Mitigation Strategies

Synchronization is of utmost importance for wireless communication as it is the key-enabler that allows communication between distant devices operating on practical, non-ideal hardware. Synchronization is necessary mainly for two reasons.

Firstly, the properties of a received radio wave depend on various environmental conditions, such as the propagation channel and the mobility of transmitter (TX) and/or receiver (RX) (Doppler effect). While the first effect can be mostly compensated using filtering strategies, the Doppler effect imposes a dynamic change in frequency and phase on the received signal, which ultimately leads to a mismatch between the analog RF chain and succeeding digital processing.

Secondly, due to tolerances in the manufacturing process of analog components (e.g., local oscillator (LO), mixer, Phase-locked loop (PLL), etc.), hardware imperfections are inevitably present in the RF front end. On top of this, time-variance of the components, such as the frequency dependency of the LO on environmental conditions like temperature, or stability issues like phase jitter, impact the transmitted and received signal fundamentally. Together, the propagation characteristics and hardware imperfections perturb the received IQ-samples in a multitude of ways, which can be summarized in four critical impacts that are explained in the following subsection.

A. *Critical Signal Perturbations*

Fig. 1 summarizes the four impacts. The left and right graphs show the transmitted (ideal) and the received (perturbed) RF signal, respectively, using 16-QAM modulation in baseband. The four centered graphs visualize each impact as they accumulate from start to end of the signal in an isolated fashion, with the insets denoting the changes observed in the constellation diagram. The red shaded region denotes the difference between the ideal and the perturbed signal. The effects and their origins are briefly explained in the following, ordered from simple to hard w.r.t. mitigation strategy complexity.

a) Sample Time Offset (STO): Time offset of the RX's Analog-to-Digital converter's (ADC) sampling times w.r.t. the matched-mfiltered "pulse-centered" IQ-samples. Static STO is caused by the ADC's random initial sample time and dynamic STO is caused by a sample rate offset (see d) below).

b) Constant Phase Offset (CPO): Quasi-static phase rotation within the observed IQ-samples. CPO is caused both by the phase rotation imprinted on the signal due to propagation distance and time, as well as by the LO's random phase in both TX and RX front end.

c) Center Frequency Offset (CFO): Frequency difference between the center frequencies of the TX's and RX's RF front end. CFO is primarily caused by precision tolerances and stability issues within LOs, however, mobility scenarios invoke the Doppler effect and directly impact signal frequency.

d) Sample Frequency Offset (SFO): Fractional mismatch between the receiver ADC's sample frequency and the instantaneous rate of received symbols. SFO has a similar origin as CFO.

As shown in the constellation diagrams in Fig. 1, these impacts either deteriorate signal quality severely or even make classic signal demodulation impossible, necessitating mitigation strategies as outline in the following.

B. *Classic Synchronization Strategies*

Synchronization strategies have to be individually tailored to the specific properties of the communication system at hand, with primary factors being the modulation type, transmit or receive diversity (i.e., SISO or MIMO), interference considerations (e.g., single/multi-user systems), and the RF front end architecture. This has led to a vast proposal of DSP algorithms, from which we want to recall the most popular and fundamental software-only algorithms (e.g., no software-controlled in-hardware Phase-locked loops) to facilitate understanding of the rest of this paper.

From a high-level perspective, all synchronization strategies perform the following sub-tasks in the following or similar order.

1) Coarse carrier frequency recovery - Correct (mean) CFO on a large timescale to a few kHz precision to enable accurate IQ-sample processing on a small timescale: Popular algorithms are the Frequency Locked Loop using band-edge filters, which converts the residual energy in the modulation's excess bandwidth to a control signal, or the "Multiply-filter-divide" method, which recovers the carrier frequency using a non-linear frequency multiplier operation.
2) Symbol timing recovery - Remove time offset in sampled IQ data to achieve optimal sampling positions: Algorithms typically resemble a "software"-PLL, such as Mueller and Muller's timing recovery scheme [15], early-late or Gardner timing error detectors, and M-path Polyphase filter banks [1]. DSP-based algorithms might use interpolation to resolve inter-sample IQ-values.
3) Fine carrier frequency and phase recovery - Remove dynamic small-scale CFO and unknown phase offset to resolve ambiguity for demodulation: Closed-loop algorithms like the Costas-Loop [16] utilize a feedback-based PLL to eradicate unwanted phase offsets on a sample-by-sample basis.

The effectiveness, computational efficiency, robustness to dynamic behavior, and acquisition time (if applicable) of these procedures highly depend on the selected DSP algorithms, whose analyses are out of scope for this paper. However, various properties allow for further optimisation, e.g., increasing spectral efficiency by removing pilot symbols or decreasing acquisition time and enhancing robustness to abruptly occurring propagation effects. To address these issues we propose a novel ML-based synchronization procedure in the next section.

III. DL-based Auto-Regressive Synchronization Strategy

In this section we propose a novel pilot-less auto-regressive ML-based synchronization strategy that mitigates STO, CFO, and CPO. For simplicity we assume uni-directional continuous communication in a SISO scenario, although generalization to full-duplex MIMO systems is discussed in Sec. V.

The strategy exhibits three distinct characteristics:

- Batch-wise processing: To facilitate computationally efficient operation, IQ data is processed in batches to conform with the ML paradigm. This is in contrast to most classic algorithms as shown in Sub-sec. II-B.

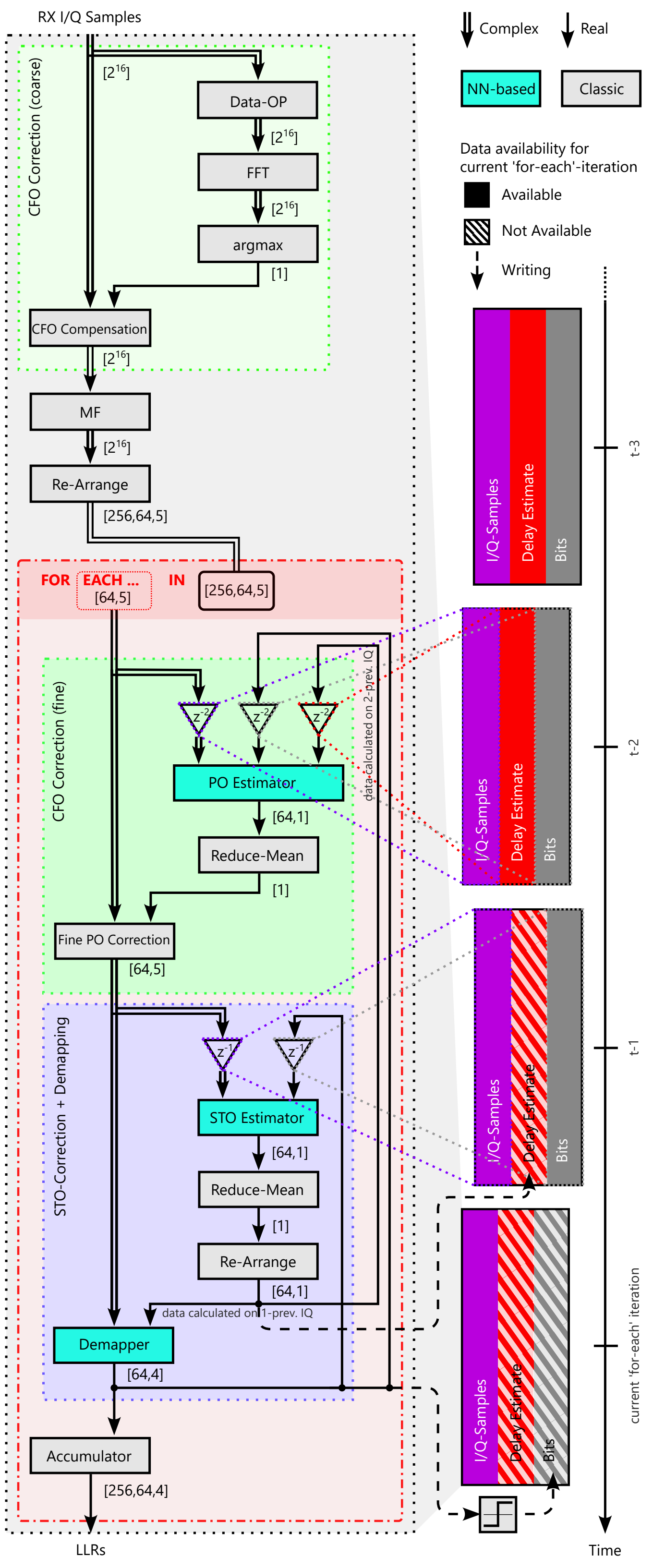

Fig. 2. Visualisation of the data flow (left) and intermediate data availability (right) in the receiver with ML-based synchronization.

- Pilot-less: The transmission of any form of known symbols (pilots) is not required, hence, spectral efficiency remains untouched.
- Auto-regressive information re-use: Information extracted from previously processed data is integrated into future processing steps.

In the following subsection we elaborate its core functionality, design choices, and data flow.

A. Algorithm Functionality and Data Flow

In a nutshell, the algorithm solves the sub-tasks listed in Sub-sec. II-B in a different order using the idea of RTNs to estimate signal properties with Neural Networks (NNs), and integrates domain knowledge by performing signal transformations with mathematical "layers". Fig. 2 visualizes the individual processing steps in a graph-like manner for transforming received IQ data to information bits. Data flows from top to bottom with the grey shaded region containing the processing pipeline for handling one batch of IQ data. The right-aligned timeline details the availability of inferred data from prior IQ mini-batches (explained below) w.r.t. time. Groups of operations that perform a distinct function, e.g., STO compensation, are highlighted by different background colors, while individual operations are either colored turquoise for ML-based operations or grey for mathematical operation. Fig. 2 assumes an IQ-batch size of 2^{16} complex samples, a mini-batch size of 256, and a receiver up-sampling factor of $\mu_{\text{RX}} = 4$. In the following we outline the implemented data flow.

The algorithm utilizes a three-step strategy to process an unsynchronized signal. In the first step (see light green shaded region), a coarse CFO compensation is performed by estimating the CFO using Fast-Fourier-Transform (FFT) on modulation-removed input data (implemented by raising the signal to the power of m), followed by a compensation via a mathematical transformation. Although FFT is used for simplicity here, convolutional neural network (CNN)-based CFO estimation is equally applicable with similar precision. Although not utilizing ML in this implementation, this step can be interpreted as an RTN with a classic feature extractor.

Before the next synchronization step, the coarsely CFO-corrected data is matched-filtered. Filtering after CFO estimation has proven to result in a higher precision for coarse CFO estimation. This is attributed to the fact that filtering of frequency-shifted data would result in an asymmetric spectrum w.r.t. the center frequency.

From this point on ML comes into play. The IQ-batch is split into equisized mini-batches which are being processed sequentially (denoted by the red shaded "for each" loop in Fig. 2). First, the phase offset in the mini-batch is estimated via means of ML and corrected using mathematical transforms. Afterwards, STO is estimated and supplied to the Demapper, both operating using NNs. The Demapper outputs Log-Likelihood-Ratios (LLRs) to allow further processing (e.g., decoding) with soft information.

B. Crucial Points of our Strategy

From an information theoretic point of view, estimation of the phase offset solely based on IQ-data is not possible due to the lack of knowledge on what happened to the signal beforehand, which is classically derived from pilot symbols. STO estimation faces a similar issue when the sampling times are centered between two symbols, leading to an ambiguity in whether to select the left or right symbol. To solve this issue we apply two creative tricks:

a) Trick 1: Both Dense Neural Network (DNN)-based estimators necessitate some kind of reference data. Providing the originally transmitted unperturbed IQ-samples as reference data enables both estimators to successfully learn producing the desired quantities; however, this data is obviously not available at the receiver. A qualified substitute is required, for which we identified the corresponding bits as an excellent alternative. By exploiting deep neural networks, feature extraction of any kind can be learned by pairing the received IQ-data with the corresponding demodulated bits. Working with these unknown bits leads to trick 2:

b) Trick 2: Although the corresponding bits are unknown for the current processed mini-batch (otherwise the task of communication would become redundant), they are acquired for previous mini-batches (assuming a running system), hence, both estimators operate on previous data, making this algorithm auto-regressive[2]. This necessitates similarity between adjacent mini-batches, which leads to trick 2: The crucial point of this strategy is to split the IQ-batch into equisized mini-batches with lengths small enough that allow assuming a similar phase offset and STO for adjacent mini-batches.

In simpler terms, we estimate CPO and STO on previously processed IQ-data combined with demodulated bits, and apply corrective transforms on the current data under the assumption of quasi-static signal characteristics.

CPO and STO are estimated symbol-wise, followed by a combine-operation (here: mean) to enhance estimation stability. Demapping is also performed symbol-wise using the "window" of corresponding IQ-samples ($\text{ups}_{\text{RX}} + 1$ to handle STO) together with the estimated STO. The final data is accumulated.

The following sub-section elaborates the training strategy for this algorithm.

C. Genie-Aided Training Strategy

Although this strategy gives rise to various hyper-parameters and architectural data handling options that require tuning, which includes but is not limited to IQ-batch size, mini-batch size, estimator output handling, characteristics of RF components, propagation properties, and mobility scenarios, we restrict our focus on the training strategy of the NNs within the algorithm for one selected operation environment only for complexity reasons.

To simplify and speed-up training we utilize a genie-aided training strategy that resolves looping over mini-batches and

[2]The phase estimator is additionally supplied with the STO estimated in the previous batch, which boosts its accuracy but leads to a $2\times$ delay.

Algorithm 1 Genie-Aided NN Training

```
Input:  Number of Epochs $E$
        Batch size $B := 1$
        Num. of IQ-Symbols per Batch $N_\mathrm{I}$
        Num. of IQ-Symbols per Mini-Batch $N_\mathrm{O}$
        Max-Frequency Offset $f_\mathrm{CFO,max}$
        Modulation-Order $m$
        Training SNR $SNR$
Output: Trained NN
 1: repeat for $E$ Epochs
 2:     Initialization:
 3:     $\mathbf{B} \leftarrow \mathrm{randBinary}([B, N_\mathrm{I}, N_\mathrm{O}, m])$
 4:     $\vec{\tau}_\mathrm{sto} \leftarrow \mathrm{randUniform}(-0.5, 0.5, [N_\mathrm{I}]) \cdot T_\mathrm{samp}$
 5:     $\phi_\mathrm{po} \leftarrow \mathrm{randUniform}(0, 2\pi)$
 6:     $f_\mathrm{cfo} \leftarrow \mathrm{randUniform}(-f_\mathrm{CFO,max}, f_\mathrm{CFO,max})$
 7:     $\mathbf{S} \leftarrow$ NN-Modulator($\mathbf{B}$) $\in \mathbb{C}^{B \times N_\mathrm{I} \times N_\mathrm{O} \times 1}$
 8:     $\vec{U} \leftarrow$ Flatten and Upsampling by $\mu_\mathrm{TX}$ of $\mathbf{S}$
 9: De-Synchronizing TX signal:
10:     for $i \leftarrow 0$ to $N_\mathrm{I}$ do
11:         $\vec{IQ}^{\mathrm{sto}}_{(i:(i+1))\cdot N_\mathrm{O}} \leftarrow \vec{U}_{(i:(i+1))\cdot N_\mathrm{O}} \circledast g_\mathrm{rcc}(t - \vec{\tau}_\mathrm{sto,i})$
12:     end for
13: Apply AWGN to set $SNR$:
14:     $\vec{IQ}^{\mathrm{Noise}} \leftarrow \vec{IQ}^{\mathrm{sto}} + \mathcal{CN}(0, \sigma^2)$
15:     $\vec{IQ}_\mathrm{RX} \leftarrow \vec{IQ}^{\mathrm{Noise}} \times e^{j \cdot 2\pi \cdot f_\mathrm{cfo} \cdot \vec{t} + \phi_\mathrm{po}}$
    Receiver Procedure:
16:     $\hat{f}^{\mathrm{coarse}}_\mathrm{cfo} \leftarrow \underset{f}{\mathrm{argmax}}(\mathrm{FFT}(\vec{IQ}^{m}_\mathrm{RX}))/m$
17:     $\vec{IQ}^{\mathrm{a}} \leftarrow \vec{IQ}_\mathrm{RX} \times e^{-j \cdot 2\pi \cdot \hat{f}^{\mathrm{coarse}}_\mathrm{cfo}}$
18:     $\vec{IQ}^{\mathrm{a}} \leftarrow \vec{IQ}^{\mathrm{a}} \circledast g_\mathrm{rcc}(t)$
19:     $\hat{\phi} \leftarrow$ EstimatePhaseOffsets($\vec{IQ}^{\mathrm{a}}, \mathbf{B}, \vec{\tau}_\mathrm{sto}$)
20:     for $i \leftarrow 0$ to $N_\mathrm{I}$ do
21:         $\vec{IQ}^{\mathrm{b}}_{(i:(i+1))\cdot N_\mathrm{O}} \leftarrow \vec{IQ}^{\mathrm{a}}_{(i:(i+1))\cdot N_\mathrm{O}} \times e^{-j \cdot \mathrm{angle}(\hat{\boldsymbol{\phi}}_\mathrm{i})}$
22:     end for
23:     $\hat{\boldsymbol{\tau}} \leftarrow$ EstimateSampleTimeOffsets($\vec{IQ}^{\mathrm{b}}, \mathbf{B}$)
24:     $\mathbf{IQ}^{\mathrm{b}} \leftarrow$ Re-Arrange $\vec{IQ}^{\mathrm{b}}$ to $[B, N_\mathrm{I}, N_\mathrm{O}, m+1]$
25:     $\hat{\mathbf{B}} \leftarrow$ NN-Demodulator($\mathbf{IQ}^{b}, \hat{\boldsymbol{\tau}}$)
26:     $l_\mathrm{bce} \leftarrow$ BinaryCrossEntropy-Loss($\mathbf{B}, \hat{\mathbf{B}}$)
27:     $l_\mathrm{custom} \leftarrow l_\mathrm{bce} + 5 \cdot \mathrm{MSE}(\vec{\tau}_\mathrm{sto}, \hat{\boldsymbol{\tau}})$
28:     Update all NNs using $l_\mathrm{custom}$
29: until
```

handling of previous data, while maintaining the algorithm's auto-regressive character. The training strategy is detailed in Algorithm 1. Algorithms 2 and 3 denote the CPO and STO estimation steps, respectively.

In summary, the algorithm is trained end-to-end within an auto-encoder-based setup. Signal perturbations are introduced on the transmitted IQ samples, i.e., CFO with random intensity per epoch sampled between $\pm f_\mathrm{CFO,max} = 50\mathrm{kHz}$, CPO with random initial phase per epoch, and STO with random offset per epoch. Phase coherency is guaranteed between epochs. Additionally, additive white Gaussian noise with constant $E_b/N_0 = 10$ dB is applied. The total training time on an NVIDIA Tesla V100 (AWS Cloud instance) for 100k epochs is approx. 5 hours. The general training mechanism is similar as in [17].

Algorithm 2 EstimatePhaseOffsets

```
Input:  IQ-Samples $\vec{IQ} \in \mathbb{C}^{B \cdot N_\mathrm{I} \cdot N_\mathrm{O} \cdot \mu_\mathrm{TX}}$
        Bit-Tensor $\mathbf{B} \in \{0,1\}^{B \times N_\mathrm{I} \times N_\mathrm{O} \times m}$
        Sample-Time-Offsets $\vec{\tau}_\mathrm{sto} \in \mathbb{R}^{B \times N_\mathrm{I}}$
Output: Estimated Phase Offsets $\hat{\phi} \in \mathbf{R}^{B \times N_\mathrm{I}}$
 1: $\mathbf{IQ} \leftarrow$ Re-Arrange $\vec{IQ}$ to $[B, N_\mathrm{I}, N_\mathrm{O}, \mu_\mathrm{TX} + 1]$
 2: $\boldsymbol{\tau} \leftarrow$ Repeat $\vec{\tau}_\mathrm{sto}$ to $[B, N_\mathrm{I}, N_\mathrm{O}, 1]$
 3: $\tilde{\boldsymbol{\phi}} \leftarrow$ NN(stack $\mathbf{IQ}$, $\mathbf{B}$, $\boldsymbol{\tau}$) $\in \mathbb{C}^{B \times N_\mathrm{I} \times N_\mathrm{O} \times 1}$
 4: $\tilde{\boldsymbol{\phi}}^{0} \leftarrow \tilde{\boldsymbol{\phi}}/|\tilde{\boldsymbol{\phi}}|$
 5: return Mean on last 2 dim of $\tilde{\boldsymbol{\phi}}$
```

Algorithm 3 EstimateSampleTimeOffsets

```
Input:  IQ-Samples $\vec{IQ} \in \mathbb{R}^{B \cdot N_\mathrm{I} \cdot N_\mathrm{O} \cdot \mu_\mathrm{TX}}$
        Bit-Tensor $\mathbf{B} \in \{0,1\}^{B \times N_\mathrm{I} \times N_\mathrm{O} \times m}$
Output: Estimated Sample Time Offsets $\hat{\tau} \in \mathbf{R}^{B \times N_\mathrm{I}}$
 1: $\mathbf{IQ} \leftarrow$ Re-Arrange $\vec{IQ}$ to $[B, N_\mathrm{I}, N_\mathrm{O}, m + 1]$
 2: $\tilde{\boldsymbol{\tau}} \leftarrow$ NN(stack $\mathbf{IQ}$, $\mathbf{B}$)
 3: return Mean on last 2 dim of $\tilde{\boldsymbol{\tau}}$
```

IV. Evaluation

To verify functionality and performance of the proposed algorithm we perform an extensive study which is summarized in this section.

Fig. 3 provides BER curves of the developed system (red), a classically synchronized pendant using techniques listed in Sub-sec. II-B (yellow), and the theoretic baseline for a

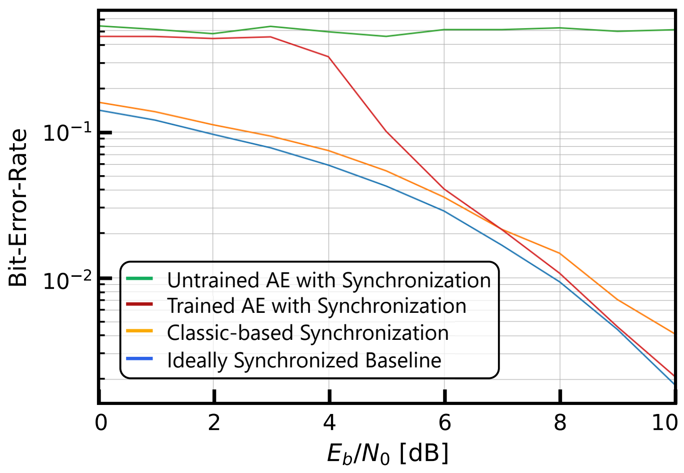

Fig. 3. Bit-Error-Rate curves for the untrained AE-based system (green), trained AE setup with AI-based synchronization (red), communication using classic synchronization techniques (yellow), and theoretic maximum performance for an ideally synchronized system (blue). A dynamic CFO with $f_\mathrm{CFO,max} = 50\mathrm{kHz}$ and varying STO is applied.

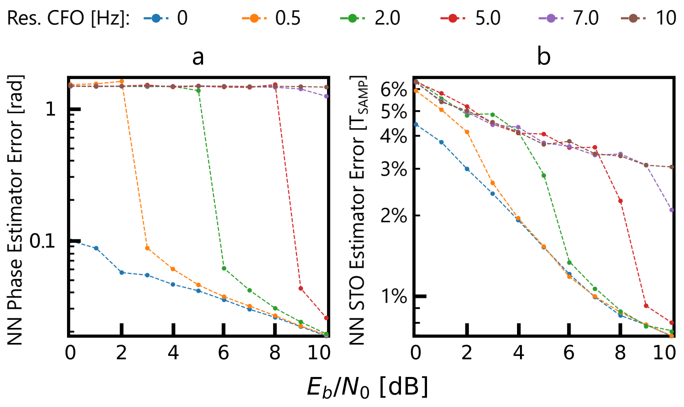

Fig. 4. Visualisation of (a) absolute NN-based phase estimation error and (b) relative NN-based STO estimation error w.r.t. the RX ADC's sampling period as a function of residual CFO and SNR. Additionally a dynamically varying STO is applied on the transmitted signal.

perfectly synchronized system without overhead (blue) as a function of E_b/N_0. It assumes 16-QAM-based transmission without coding in a realistic environment (STO and CFO vary dynamically during transmission, similarly as implemented for NN-training). It can be seen that the classic approach exhibits a constant performance gap of approx. 0.7 dB, while the AI-based synchronized setup reaches a negligible gap in the high SNR region. Nevertheless we note, that communication in the low SNR region is error-prone due to bit error propagation in the auto-regressive algorithm structure, as explained in the following.

To understand the above observed behavior and provide a more detailed analysis on the performance of the synchronization algorithm and its individual components, we study NN-based STO and CPO estimation precision as a function of CFO intensity and SNR while applying a dynamically varying STO. To simplify understanding we excluded the FFT-based coarse CFO compensation from this analysis and adapted the applied CFO range to match the residual CFO after this step.

Fig. 4(a) shows the phase estimator error, and Fig. 4(b) shows the STO estimator error. It is evident that both estimators strongly depend on both residual CFO and SNR, and that higher CFOs necessitate higher SNRs to uphold precision, which is intuitively expected. We could observe that when the SNR is low, too high residual CFOs lead to mis-estimations that fuel a chain-reaction of bit error propagation, prohibiting further communication.

V. Conclusion and Future Work

In this work we proposed, implemented, and evaluated a novel pilot-less auto-regressive RF synchronization strategy using Deep-Learning. By extending a state-of-the-art bit-wise auto-encoder-based setup we demonstrated continuous unidirectional communication by simulating a realistic environment, perturbed by hardware imperfections in the RF front end and propagation perturbations, leading to STO and CFO. Although low-SNR communication is highly error prone due to auto-regressive bit error propagation, we achieved a gain of approx. 0.6 dB in the high-SNR region.

Future work might consider improving this algorithm, e.g., by minimizing error propagation using soft information (LLRs instead of bits) within the NN-based parameter estimators, and extending this strategy to MIMO communication by deploying the algorithm on every RF path by utilizing a batch-size > 1. Although we believe that this work can serve as an introduction to ML-based synchronization, more questions have been raised than answered. To reach maturity some issues remain, such as developing a start-up procedure when no prior data is present, and extending this work by adding Sample Frequency Offset compensation.

Acknowledgment

We would like to thank Marc Lichtman for his work on PySDR and the resulting fruitful discussion.

References

[1] f. harris, *Let's Assume the System Is Synchronized*, 01 2011, pp. 311–325.

[2] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[3] I. C. Society, "Machine learning for communications emerging technologies initiative," https://mlc.committees.comsoc.org/.

[4] X. Lin, "Artificial intelligence in 3gpp 5g-advanced: A survey," 2023.

[5] W. Xu, Z. Zhong, Y. Be'ery, X. You, and C. Zhang, "Joint neural network equalizer and decoder," in *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, 2018, pp. 1–5.

[6] S. Cammerer, J. Hoydis, F. Aoudia, and A. Keller, "Graph neural networks for channel decoding," 12 2022, pp. 486–491.

[7] T. T. Vu, D. T. Ngo, N. H. Tran, H. Q. Ngo, M. N. Dao, and R. H. Middleton, "Cell-free massive mimo for wireless federated learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6377–6392, 2020.

[8] H. Xing, O. Simeone, and S. Bi, "Decentralized federated learning via sgd over wireless d2d networks," in *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5.

[9] D. Liu, G. Zhu, J. Zhang, and K. Huang, "Wireless data acquisition for edge learning: Importance-aware retransmission," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2019, pp. 1–5.

[10] T. J. O'Shea, L. Pemula, D. Batra, and T. C. Clancy, "Radio transformer networks: Attention models for learning to synchronize in wireless systems," in *2016 50th Asilomar Conference on Signals, Systems and Computers*, 2016, pp. 662–666.

[11] S. Dörner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep learning based communication over the air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, 2018.

[12] S. Cammerer, F. A. Aoudia, S. Dörner, M. Stark, J. Hoydis, and S. ten Brink, "Trainable communication systems: Concepts and prototype," *IEEE Transactions on Communications*, vol. 68, no. 9, 2020.

[13] J. Liu, K. Mei, X. Zhang, D. McLernon, D. Ma, J. Wei, and S. A. R. Zaidi, "Fine timing and frequency synchronization for mimo-ofdm: An extreme learning approach," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 2, pp. 720–732, 2022.

[14] J. Clausius, S. Dörner, S. Cammerer, and S. T. Brink, "On end-to-end learning of joint detection and decoding for short-packet communications," in *2022 IEEE Globecom Workshops (GC Wkshps)*, 2022, pp. 377–382.

[15] K. H. Mueller and M. Muller, "Timing recovery in digital synchronous data receivers," *IEEE Trans. Commun.*, vol. 24, pp. 516–531, 1976.

[16] J. P. Costas, "Synchronous communications," *Proceedings of the IRE*, vol. 44, no. 12, pp. 1713–1718, 1956.

[17] M. Petry, A. Koch, and M. Werner, "Envisioning physical layer flexibility through the power of machine-learning," in *2023 IEEE Globecom Workshops (GC Wkshps)*, "in press".