Trajectory-Unaware Path Loss Forecast in a Distributed Massive MIMO System based on a Multivariate BiLSTM Model

Rodney Martinez Alonso 1, Robbert Beerten1, Achiel Colpaert1,2, Andrea P Guevara1, and Sofie Pollin1

1Department of Electrical Engineering (ESAT)
2IMEC

February 19, 2024

Abstract

Cell-free massive MIMO networks have recently emerged as an attractive solution capable of solving the performance degradation at the cell edge of cellular networks. For scalability reasons, user-centric clusters were recently proposed to serve users via a subset of APs. In the case of dynamic mobile scenarios, this form of network organization requires predictive algorithms for forecasting propagation parameters to maintain performance by proactively allocating new APs to a user. In this paper, we present a BiLSTM-based multivariate path loss forecasting algorithm. Thanks to the combination of dual prediction by the BiLSTM and diversity from multiple antennas, our model mitigates the error propagation typically faced by sequential neural networks for time-series forecasting. In the evaluated scenario, from 2 to 10 steps ahead, we reduce the propagation of the error by a factor of 18 compared to previous research on path loss forecasting by an LSTM time-series-based model. In contrast to parallel transformer solutions, the complexity cost of our algorithm is also significantly lower.
ABSTRACT  Cell-free massive MIMO networks have recently emerged as an attractive solution capable of solving the performance degradation at the cell edge of cellular networks. For scalability reasons, user-centric clusters were recently proposed to serve users via a subset of APs. In the case of dynamic mobile scenarios, this form of network organization requires predictive algorithms for forecasting propagation parameters to maintain performance by proactively allocating new APs to a user. In this paper, we present a BiLSTM-based multivariate path loss forecasting algorithm. Thanks to the combination of dual prediction by the BiLSTM and diversity from multiple antennas, our model mitigates the error propagation typically faced by sequential neural networks for time-series forecasting. In the evaluated scenario, from 2 to 10 steps ahead, we reduce the propagation of the error by a factor of 18 compared to previous research on path loss forecasting by an LSTM time-series-based model. In contrast to parallel transformer solutions, the complexity cost of our algorithm is also significantly lower.

INDEX TERMS  Channel Prediction, Path Loss Forecast, Mobile Networks, Massive MIMO, Machine Learning, Bidirectional LSTM.

I. INTRODUCTION

SINCE the first generation of mobile networks, we have relied on different variations of the cell-based network architecture. Although the cell-based infrastructure allows better management of the network quality of service compared to distributed unlicensed technologies, it comes at a cost of lower flexibility and performance degradation in the cell edges [1]. Inter-cell interference and its impact on the quality of service, particularly at the cell edge, has been a key limitation for further improvement of user-perceived performance. All multiplexing schemes that aim to optimize user performance by exploiting space, time, frequency, and coding as the system’s degrees of freedom, eventually fail at the cell edge as they are limited by inter-cell interference [2].

It is fair to say that, despite its flaws, the cell-based infrastructure organization has been successful in terms of network resource management. However, within the constraints of the traditional single-cell-based topology, there is no wide margin for solving the cell-edge performance gaps. For instance, a marginal improvement of barely 10% cell-edge throughput is typically achieved by single-cell-based optimization algorithms [1]. Indeed, in [3] authors demonstrated that for mitigating the inter-cell interference a multi-cell cooperation approach is required. For example, this proposal considers a cell-based massive MIMO with partial multi-cell minimum-mean square error combining vector for reducing the interference compared to the traditional single-cell optimization [3].
Despite the technological challenges to solve, cell-free massive MIMO has raised as a major paradigm change in the traditional infrastructure topology of the mobile networks [4]. Rather than self-competing for the radio resources, the cell-free MIMO topology allows for exploiting the spatial diversity from multiple antennas and Access Points (APs). Distributed Access Points allow us to overcome cell-edge performance degradation and offer macro-diversity against shadowing [5]. In this infrastructure topology, the APs jointly collaborate to serve all users in their reach, offering a ubiquitous connectivity solution [6]. Nevertheless, the traditional canonical cell-free approach of all APs serving all users has limited feasibility for large-scale networks, due to high computational and front-haul requirements [7]. One of the key technological challenges to solve is indeed finding the best trade-off between the computational complexity, front-haul requirements, and the advantages of a cell-free topology in terms of reduced interference, and higher spectral and energy efficiency [7]. User-centric clustering, where the network dynamically creates clusters of collaborative APs serving each user, has been proposed for decreasing the amount of signalization information and improving the ratio between overhead and effective payload [8].

Most research literature considers a static cluster formation where, once the cluster is created, it is assumed the APs’ assignment remains static while the communication lasts [7], [9]–[14]. However, in a mobile scenario, the channel does not remain stationary as the user moves or the environment changes [15]. As a consequence, the cluster of APs serving a user will dynamically change over time as the path loss perceived by each AP varies. This means the cluster of APs serving a user has a limited lifetime dependent on the perceived path loss as the user moves. Therefore, developing path loss forecasting models is fundamental for cluster predictive reformation in a realistic mobile environment. The specific problem we aim to investigate is the tradeoff between the accuracy and complexity of forecasting algorithms for predicting path loss evolution in distributed massive MIMO systems.

Mobility prediction strategies have focused mainly on cell-based infrastructure. For 5G networks prediction algorithms have proven to reduce the performance drops during handover. At the same time, predictive channel evolution can reduce the amount of overhead and signalization used for channel estimation [16], [17]. For instance, authors in [16] investigated a Long-Short-Term-Memory (LSTM) neural network for proactively predicting the handover and effectively reducing the handover cost function. However, a critical limitation is that the algorithm requires a significant backup of radio resources for the handover, that remains unused otherwise. For a more efficient radio resource management, authors in [17] also investigated an LSTM-based algorithm for the proactive prediction of the handover based on the user trajectory prediction. In their model, the algorithm is capable of predicting the 2D location of the users in future time steps.

User mobility and trajectory prediction have been widely investigated in the literature. With the recent advances in artificial intelligence and machine learning, different techniques like Recurrent Neural Networks (RNNs) for time series forecasting have been widely used for this purpose [18], [19]. Nevertheless, in most realistic mobile network scenarios the spatial information of the user is unlikely to be known while inferring it will have a significant computational burden and impact in the effective payload of the front-haul [20], [21].

For solving the shortcomings associated with the availability of spatial data (i.e., users’ location \( X; Y \)) in the prediction of the path loss evolution caused by user mobility, some papers have investigated the use of different types of sequential algorithms [22]–[26]. The main limitation with the typical RNN and other sequential processing machine learning algorithms for time series forecasting is that they can only predict the channel gain in the next time stamp \( t + 1 \) based on the whole evolution history. For further predictions \( t + L, L > 1 \) they need to combine iteratively the previous evolution history with the current estimate to predict a next time stamp. As the channel is sequentially predicted in an iterative closed-loop, the error tends to propagate in an accumulative way over the time series [27]. Some preliminary results were obtained by authors in [28] on maximizing the accuracy in the sequential prediction of the channel. Their solution depicts a combined LSTM and Convolutional Neural Network, enhancing the temporal channel gain prediction by considering not only the past and present state but also the information from their spatial neighbors. Although the proposed model significantly improved the prediction accuracy by a factor of 1.5 to 2 compared to other RNN and LSTM models, the error propagates at a similar rate when the number of steps to be predicted increases [28]. Nevertheless, the perspective of collaborative sensing by multiple APs could be worth exploring in a distributed massive MIMO system, for maximizing the prediction accuracy without labeled positioning data.

From our literature review, one of the most relevant research in the state-of-the-art on path loss forecasting in massive MIMO is based on a neural network transformer model [27]. The authors present a parallel transformer-based prediction of the channel, capable of predicting several L-frames in the future. Their model achieved a higher accuracy of the future channel prediction compared to the traditional LSTM, mitigating also the error propagation across multiple time stamps. Indeed, from [28] and our own experimental assessment, the traditional LSTM time series forecast only achieves a high accuracy for \( L = 1 \). However, the computational complexity of the attention mechanism required for the parallel transformer-based model in [27] has a quadratic dependence on the number of antennas, and the length of the path loss sequence.
To achieve a better trade-off between the model complexity, path loss prediction accuracy, and mitigating the propagation of the error, we propose a Bidirectional LSTM (BiLSTM) neural network capable of exploiting the spatial diversity of a distributed massive MIMO architecture. Notice that, for the evaluation of our algorithm the architecture, either cell-free or distributed massive MIMO, is transparent as we only use the channel state information from all antennas in range. The BiLSTM neural network maximizes the accuracy by a joint forward and backward prediction of the path loss across all the distributed antennas. We hypothesize that our model is capable of achieving a similar or higher accuracy than [28] and [27], with a significantly less complex architecture. As the forward and backward sub-layers will predict the same value for the middle of the time series (i.e., hybrid prediction as the time series is mirrored from its middle), the propagation of large errors is less likely. Therefore, we hypothesize that for predicting multiple steps in a closed loop, our model will be capable of limiting the propagation of the error (i.e., self-correcting).

Our main contributions can be summarized as follows:

- We developed a sequential channel prediction model based on a BiLSTM, without labeled positioning data, and capable of predicting the path loss for several time stamps ahead from its historical evolution, achieving a higher accuracy compared to the state-of-the-art.
- The combination of distributed massive MIMO spatial diversity and the hybrid forward and backward prediction of BiLSTM allow for limiting the propagation of the error across multiple future predictions. Indeed, our proposed model significantly reduces the error propagation compared to traditional time-series prediction models.
- Compared to fully parallel transformer-based solutions, our model does not require an attention mechanism and has a significantly smaller computational complexity.

The outline of this paper continues as follows: In Section II, we present our BiLSTM-based path loss forecasting algorithm for distributed massive MIMO, including the details of the generation of the path loss evolution dataset (Subsection A), the core algorithm for the time series forecasting and neural network architecture (Subsection B), and the multivariate model for inferring the path loss for several time stamps ahead (Subsection C). Our research findings and detailed numerical analysis, focusing on the influence of the sequence length, dataset size, antenna configuration, number of BiLSTM cells required, and time steps to predict, are presented in Section III. Finally, our research conclusions are highlighted in Section IV.

II. METHOD

The one-slope path loss model allows generalizing the attenuation experienced by the signal as a function of the distance between a transmitter and receiver for a certain frequency and given scenario, accounting for propagation loss measurements in a certain scenario. Formally, the propagation loss between one AP and the user is described as:

$$\beta[dB] = \beta_0 + 10 \cdot \nu \cdot \log\left(\frac{d}{d_0}\right) + \chi \quad (1)$$

where $d$ is the distance between the transmitter and the receiver, $d_0$ is a reference distance for the model, $\beta_0$ [dB] is the path loss at the reference distance $d_0 \ll d$, $\nu$ is the path loss exponent, $\chi$ represents a normal distribution of the large-scale and small-scale fading. In Equation 1 the spatial linked components of the path loss (i.e., distance-dependent) will match the mean path loss $\beta$ (i.e., $50^{th}$-percentile). Here, we consider only the losses in the channel between the transmitter’s antenna output and antenna input at the receiver. Our aim is to create a model capable of forecasting the mean path loss without any location information but relying on the previously recorded path losses for different time stamps (i.e., matching to $\beta = \beta_0 + 10 \cdot \nu \cdot \log(d/d_0)$ in Equation 1) [29].

The traditional LSTM allows for the modeling of the time series with a memory dependence. This is performed by looping over each time step in the series and updating the RNN state. For forecasting future steps of the path loss data sequence the previous step is used as input. Therefore, the input sequence is the same as the output shifted left by one timestamp. In this paper, we first investigate openloop forecasting, where only the historical data is used for predicting the next time stamp. Further, we will investigate a multivariate model for predicting $L$-steps ahead based on closed-loop forecasting. For this case, the prediction of each time stamp in the sequence uses the previous prediction as input.

For improving the computational efficiency, and accuracy in forecasting the path loss without spatial labeled data, we propose a method based on a BiLSTM neural network capable of exploiting the spatial diversity of a distributed array of antennas. For evaluating our algorithm we adapted the static dataset of a distributed massive MIMO testbed presented by [30], [31] to generate a mobility dataset. In the following subsections, we provide details of the synthetic mobility dataset generation and the BiLSTM algorithm designed.

A. PATH LOSS EVOLUTION DATASET GENERATION

For evaluating our neural network performance we modified the distributed ultra-dense Channel State Information (CSI) dataset from KU Leuven Massive MIMO testbed measurements presented in [30]–[32]. This dataset comprises the recorded complex CSI data by 64 antennas distributed among 8 linear arrays (i.e., APs). The CSI is precisely labeled to a cartesian reference coordinate in the spatial domain with a resolution of 5 mm [30], [32]. In our application, this ultra-dense spatial resolution is not required. Therefore we downsampled the dataset to an equivalent resolution of 5 cm. As we aim to abstract from the spatial domain data, the posi-
tioning labels were removed during the data pre-processing. The transmission from the user is a 20 MHz OFDM signal with a center frequency of 2.61 GHz ($\lambda = 0.11456$ m). The CSI was recorded for 100 sub-carriers [30], [32]. For the evaluated scenario, there is a strong similarity of the measurements in the frequency domain. Therefore, for the intended application we accounted for the mean path loss over all the sub-carriers.

As this is a static-based CSI dataset, we emulated the path loss variation caused by the user mobility as a multi t-steps dependent probabilistic chain. Therefore, the path loss variation ($\Delta \beta$) caused by a transition from the user position $[X_A(t); Y_A(t)]$ to the position $[X_B(t+1); Y_B(t+1)]$ at a given time variation $\Delta t$ can be modeled as a stochastic time-dependent process. In this case, the transition from the state $A$ to the state $B$ is determined not only by the current and past state of the system (constrained degrees of freedom), but also by an element of randomness. Nevertheless, some transitions between certain neighboring states have a higher likelihood than others.

As one of the main limitations identified in the literature is the link between the objective variable (e.g., path loss) and the spatial domain data (i.e., $X; Y$), we modeled the path loss variation without the positioning labels. Each possible state will correspond to a vector comprising the path loss variations without the positioning labels. Each possible state will uniquely be identified from the combination of the sensed channel state by at least three APs. Therefore, if two mobility speed of approximately $1 \text{ m/s}$, the similarity factor between any given $\beta_i$ vector and their cluster centroid.

Therefore, each $\beta_i$ vector will be assigned to the closer cluster in terms of its sum squared geometrical distance from the cluster centroid. In the second step (Maximization), the objective variable is differentiated over the centroids. Therefore, the clusters are redesigned by recomputing the centroids and a new cluster assignment:

$$\frac{\partial J}{\partial C_k} = 2 \sum_{i=1}^{N_D} \omega_{ik} (\beta_i - C_k) = 0$$
$$\Rightarrow C_k = \frac{\sum_{i=1}^{N_D} \omega_{ik} \cdot \beta_i}{\sum_{i=1}^{N_D} \omega_{ik}}.$$ (4)

From the K-mean classifier the neighboring $\beta_i$ vectors will be grouped depending on their similarity up to a maximum geometric distance $R$ over the normalized data, been $r < R$ the similarity factor between any given $\beta_i$ vector and their cluster centroid.

Finally, the path loss variation caused by the mobility can be represented by a pseudo-random transition from one cluster to another. Once the $K$ clusters of $\beta_i$ vectors are created, we generate $N_s$ different path loss evolution sequences of length $Z$ (i.e., timesteps) by a pseudo-random generation of transitions between clusters. We need to clarify that a purely random behavior of the user can not be modeled or predicted, but a certain pattern. Therefore, we vectored the transitions between neighboring clusters constraining the degree of freedom to any $\beta_i$ within a neighboring cluster that is not already in the sequence.

In Fig. 1 we show a visual representation of the generated path loss evolution sequences. The picture depicts one path loss evolution sequence (e.g., $S_t$) of the total possible $N_s$ sequences, with a memory length $Z$, corresponding to the emulation of a certain trajectory in the experimental scenario. Here, the historic evolution on any given sequence is assumed as the path loss perceived by all the access points across different time stamps $t$, e.g., $S_t \in \mathbb{R}^{N_s \cdot Z}$ in Fig. 1. Therefore, the generated dataset of path loss evolution sequences, that will be used for evaluating the forecasting algorithm in the following subsections, will consist of a tensor defined as:

$$S \in \mathbb{R}^{N_s \cdot N_{AP} \cdot Z}.$$ (5)

B. BILSTM-BASED OPEN-LOOP PATH LOSS FORECAST

Algorithm 1 describes the machine learning algorithm for forecasting the path loss for each AP at any given time stamp ahead $[\beta_i, t + 1]$ based on a certain user path loss evolution history.

**Inputs:** The algorithm receives as input the tensor containing the path loss evolution sequences $S \in \mathbb{R}^{N_s \cdot N_{AP} \cdot Z}$ generated as described in subsection A. For evaluating the machine learning performance and accuracy we will assess the influence of the length of each sequence (historic data in the time domain) for $Z = \{2, 5, 8, 10, 15, 20\}$, the dataset
Algorithm 1 Machine learning algorithm based on a BiLSTM open-loop for forecasting the user’s path loss perceived by each AP.

Input: $S \in \mathbb{R}^{N_s \times N_{AP} \times Z}$
Output: $\beta[t + 1] \in \mathbb{R}^{N_{AP}}$

1: $t_{train} = 1 : \lceil (0.9 \times N_s) \rceil$;
2: $t_{test} = \{t_{train} + 1 : N_s\}$;
3: $\text{dataTrain} = S[t_{train}, :, :]$
4: $\text{dataTest} = S[t_{test}, :, :]$
5: for $n = 1 : \text{numel(dataTrain)}$
6: $X = \text{dataTrain}(n)$;
7: $XTrain[n] = X(:,1:end-1)$;
8: $TTrain[n] = X(:,2:end)$;
9: end for;
10: for $n = 1 : \text{numel(dataTest)}$
11: $X = \text{dataTest}(n)$;
12: $XTest[n] = X(:,1:end-1)$;
13: $TTest[n] = X(:,2:end)$;
14: end for;
15: Normalize ($XTrain,TTrain$)
16: Normalize ($XTest,TTest$)
17: [net,info] = $\text{trainNetwork}(XTrain,TTrain,\text{layers,options})$
18: $YTrain = \text{predict}(\text{net},XTrain)$
19: $YTest = \text{predict}(\text{net},XTest)$
20: $\text{RMSE}(YTrain,TTrain)$
21: $\text{RMSE}(YTest,TTest)$
22: $\beta[t + 1] = YTest[t]$
at each time stamp. This long-term memory learning leads to maximizing the accuracy and performance of the model. Without being a limitation, it is important to mention that the sequential input needs to be buffered in batches at least equal to the minimum sequence size. Equations 7 and 8 describe the functional model of the BiLSTM layer in our application for the forward and backward LSTM sub-layers, respectively.

\[
\left\{ \hat{\beta}^t [\cdot; t - (Z - 1)]; \ldots; \hat{\beta}^t [\cdot; t - \left( \frac{Z}{2} + 1 \right)]; \ldots; \hat{\beta}^t [\cdot; t] \right\} = \overline{F}_w \left( \beta^t [\cdot; t - (Z - 2)]; \ldots; \beta^t [\cdot; t - \left( \frac{Z}{2} \right)]; \ldots; \beta^t [t - 1] \right) \quad (7)
\]

\[
\left\{ \hat{\beta}^t [\cdot; t]; \ldots; \hat{\beta}^t [\cdot; t - \left( \frac{Z}{2} + 1 \right)]; \ldots; \hat{\beta}^t [\cdot; t - (Z - 1)] \right\} = \overline{B}_w \left( \beta^t [\cdot; t - 1]; \ldots; \beta^t [\cdot; t - \left( \frac{Z}{2} \right)]; \ldots; \beta^t [\cdot; t - (Z - 2)] \right) \quad (8)
\]

The forward and backward training of a batch of the time-series allows for reducing the uncertainty caused by the last sample of the sequence as no future data is available (present time effect). In the BiLSTM architecture, all elements of the sequence can be used for the training of the model either as a past or future time step. In this way, for each time stamp its future and past state can be used for minimizing the loss function of the model, improving its accuracy and convergence. Nevertheless, the optimizer will tend to maximize the weights fine-tuning accuracy for the middle of the series where both sub-layers should predict the same future step [34], i.e., \( \hat{\beta}^t \left[ t - \left( \frac{Z}{2} + 1 \right) \right] \) would be predicted from \( \hat{\beta}^t \left[ t - \left( \frac{Z}{2} \right) \right] \) in both the forward and backward LSTM sub-layer. This feature will be particularly relevant for the multivariate model closed-loop forecasting that we present in the following subsection.

We will investigate the optimal number of BiLSTM cells in the range of \( \{2, 4, 8, 16, 32, 64, 128, 256, 512\} \). In the fully connected layer, the number of neurons is set to the size of the batch multiplied by the sequence length \( Z \). For a fair comparison, we trained each model at least for 1000 iterations, at a learning rate of 0.01, and assuming a mini-batch size of 32. For this combination of hyper-parameters the maximum reachable accuracy of the model is guaranteed, i.e., the progressive average of the loss function converges to a constant value. Nevertheless, after the best model is selected a further fine-tuning of the machine learning hyper-parameters is realized for achieving a better trade-off between accuracy and computational performance.

It is fundamental to avoid generating an over-fitted model. This means, avoiding a model with high accuracy in the training data, but poor performance for unknown sequences (testing subset). For this, we evaluate the capability of a Dropout layer to avoid generating an over-fitted model. During the training phase, the dropout layer will randomly drop connections to neural units within a certain probability \( P_{drop} \). By skipping such connections the optimizer will avoid an excessive weight co-adaptation to the training data. This can be seen as adding a certain noise to the data during training, increasing the tolerance of the model to small features that might be irrelevant and therefore will likely not match another batch of data.

Once the BiLSTM neural network model is trained it will allow predicting the path loss for all the APs at the next future time stamp \( \hat{\beta}^t [\cdot; t + 1] \) with a certain accuracy. For estimating the model forecasting accuracy, we compute the Root-Mean-Squared Error (RMSE) considering both the subset of data used for the training and the reserved subset for testing, from which the model has no previous knowledge (line 18 to 22 in Algorithm 1). Equation 9 describes the general function accounting for the RMSE across all the predictions over any element \( i \) of each AP within each sequence in the subset \( N_s \) of length \( Z \).

\[
RMSE = \sqrt{\frac{1}{N_s \cdot N_{AP} \cdot Z} \sum_{i=1}^{N_s \cdot N_{AP} \cdot Z} (\beta_i^t - \hat{\beta}_i^t)^2} \quad (9)
\]

where \( \beta_i^t \) and \( \hat{\beta}_i^t \) respectively correspond to the actual and forecasted path loss for the \( i^{th} \) element in the dataset.

C. MULTIVARIATE CLOSED-LOOP PATH LOSS FORECAST

For predicting the path loss \( L \)-steps ahead, we modified the inference part after Algorithm 1 as a closed-loop multivariate prediction model. The BiLSTM-based architecture will remain the same, but the feed of the data during the inference part will be modified. Algorithm 2 describes the closed-loop multivariate inference model for sequentially forecasting \( L \)-steps ahead in the time series.

After the neural network model is generated (Input \( \text{net} \) in Algorithm 2), during the inference phase the algorithm receives as input the subset of data reserved for testing. In this case, for predicting each subsequent time stamp of the time series, the previous prediction will be used as input (i.e., recursive closed-loop forecasting). Therefore, at each iteration \( t \) the predicted values in \( Z_{pred} \) are fed as the current stamp for the next iteration, and the model is accordingly updated (line 1 to 4 in Algorithm 2). After \( L \) iterations all
Algorithm 2 Multivariate Closed-Loop Path Loss Forecasting.

Input: net, XTest \( \subset S \in \mathbb{R}^{N_s, N_{AP}, Z} \)
Output: \( S_L \)
1: for \( t = (Z - L) : Z - 1 \)
2: \([\text{net}, \text{Zpred}] = \text{predictAndUpdateState}(\text{net}, \text{XTest},...)\)
3: \( \text{XTest}(t) = \text{Zpred}(t - 1) \)
4: end for;
5: \( S_L = \text{Zpred}; \)

the future predictions for each sequence in \( S \) are generated \((S_L)\). Although in the traditional LSTM and other RNNs, this creates an accumulative propagation of the error, this is not the case with BiLSTM. We hypothesize that because of the hybrid prediction features in the BiLSTM dual-layer architecture, large errors are unlikely to propagate across iterations. For a certain sequence, a large error caused by one of the layers will have a mirroring effect in the opposite layer after a certain number of time stamps.

III. RESULTS

A numerical analysis of the results shows that the memory depth \( Z \) of the path loss evolution history has the highest impact in terms of accuracy, as well as the density of BiLSTM cells in the neural network layer. Although the number of samples in the dataset is also related to the accuracy (i.e., not enough samples might lead to an under-representation of the propagation environment), for the evaluated scenario there is not a significant variation in the density of BiLSTM cells for the maximum achievable accuracy as the function of \( N_s \). The addition of the dropout layer has a minimal impact in terms of accuracy. However, it reduces the likelihood of an overfitted model to the training subset for further applications in an online loop. In further subsections, a detailed numerical analysis of these general findings is presented.

A. HISTORY LENGTH AND PREDICTION ACCURACY

Fig. 3 shows the accuracy of the evaluated time-series forecasting models for predicting the path loss on a sequential basis. The graphs correspond to the mean RMSE and 95th percentile root squared error in both the training subset and the testing subset for which the model has no previous knowledge of the path loss evolution. Fig. 3 corresponds to the architecture comprising the sequential input layer, the BiLSTM layer (a) with a dropout layer and (b) without it), the fully connected layer and the regression layer. Here, we consider the size of the dataset of path loss sequences of \( N_s = 100 \) and a distributed array of 8 APs by 8 antennas each. We show the best performance in terms of model accuracy, corresponding to a BiLSTM density of 256 to 512 BiLSTM cells and a dropout probability of 0.1 to 0.15.

Our results reveal that the gradient of the error significantly increases for sequence length below 10. Although for the inflection point of \( Z = 10 \) the RMSE (50th-percentile) in the testing subset is only 0.35, for the 95th-percentile of the previously unknown path loss evolution per AP, the error increases near 0.5 (in Fig. 3a). The best achieved performance corresponds to \( Z = 20 \), for which the RMSE is lower by a factor 1.5. The addition of the dropout layer barely improves the accuracy in the testing subset by a factor 1.1 for \( Z \geq 15 \). Nevertheless, as the size of the testing subset is 8 APs by 10 sequences only, for an online inference the dropout layer might be fundamental in order to reduce the likelihood of having an overfitted model to the evaluated training subset. In this regard, the dropout layer allows reducing the performance difference between the training subset and testing subset for \( Z = 20 \) from 66% to 12% for the 50th-percentile, and from 42% to barely 5% for the 95th-percentile of sequences per AP.

Fig. 4 shows the model accuracy on the training and testing subsets for different densities of BiLSTM cells for a path loss history memory depth of \( Z = 10 \) and \( Z = 20 \), without any correction from the dropout layer. A higher number of samples in the training sequence does not necessarily lead to a higher density of BiLSTM cells. For \( Z = 20 \) the model...
achieves an acceptable performance on both the training and test subset from only 64 BiLSTM cells onward. For $Z = 10$, at least 256 BiLSTM cells are required to achieve an error below 0.5 for most of the sequences in the testing subset. Notice that above an error of 0.5 not only does the error increase, but also the gradient of the predicted path loss sequence might have a significant difference for some trajectories. For an $RMSE \geq 0.59$ both the absolute value of the gradient and its sign might mismatch for some sequences. Nevertheless, below this threshold, the algorithm can at least predict the trend in $\beta$. Fig. 5 shows an example visually representing this effect.

**B. NUMBER OF PATH LOSS SEQUENCES AND PREDICTION ACCURACY**

Fig. 6 shows the model accuracy for different sizes of the path loss sequence dataset. The simulations correspond to the edge-performance case of $Z = 10$ with 256 BiLSTM cells, without, and with a dropout layer correction equivalent to a drop probability of 0.1. It is important to mention that for $N_s \leq 50$ the model is extremely over-fitted to the training subset (i.e., training to testing subset RMSE difference of 1:7). This is because the number of path loss sequences is too small for having an accurate digital twin model representation of the whole environment and possible path loss evolution. Therefore, for $N_s \leq 50$ the model standard deviation across different training of the machine learning is not enough to guarantee its reproducibility.

For $N_s \geq 100$, Fig. 6 shows a trend towards a decrease in the model error for a higher number of sequences. For instance, for $N_s = 200$ the RMSE is lower than for $N_s = 100$ by a factor of 1.2 either for $P_{drop} = 0$ or $P_{drop} = 0.1$. By reducing the over-fitting likelihood, the inclusion of the dropout layer reduces the mean error in the testing subset at least by 4% (i.e., for $N_s = 200$).

**C. INFLUENCE OF ANTENNAS CONFIGURATION**

We compared the accuracy of the distributed antenna array in an 8-by-8 configuration to the prediction for a single AP with 8 antennas. Fig. 7 shows the results of this comparison considering 256 BiLSTM cells, $N_s = 100$, and without any correction by the dropout layer.

The analysis of Fig. 7 reveals that although for longer path loss sequences $Z > 8$ there is not a significant difference for the $50^{th}$-percentile of the sequences, the model for the single array has a higher deviation from the mean. For the $95^{th}$-percentile the 8-by-8 AP array has an error lower by a factor of 1.6 (i.e., for $Z = 20$). For the edge performance case of $Z = 10$ the error of the multi-APs serving configuration is lower by a factor of 1.8. Moreover, notice that in the case of the single AP, the error can be higher than 0.8, which means that for some path loss sequences the model
can not provide an accurate representation of any feature of the path loss evolution. A visual representation of the improved accuracy by the combined prediction of the 8 APs compared to a single AP can be noticed from Fig. 1. For instance, in Fig. 1 the abrupt change in the path loss for AP$_2$ at the end of the sequence is unlikely to be learned by the BiLSTM as it does not match most of the sequence evolutions. However, the BiLSTM neural network can easily learn the fingerprinting feature from the combined evolution across $S_1 \{ \beta'_{AP_1}, \ldots, \beta'_{AP_8} \}$.

**D. MULTIVARIATE CLOSED-LOOP PREDICTION ACCURACY**

Fig. 8 shows the prediction accuracy for the multivariate closed-loop model for sequentially predicting $L$-path-loss steps ahead, considering the path loss history for different sequence length $Z$.

As with the single prediction model, the multivariate model has an accuracy dependency on the path loss sequence length. For $Z = 20$ it is possible to accurately predict around 14 time steps ahead, while for $Z = 10$ this is barely 2 time steps ahead. For each of the evaluated sequence lengths, the error increases with the number of steps to predict in advance. For instance, for $Z = 20$ and $L = 14$ the mean error is 1.2 times higher than for $L = 4$. A similar trend occurs for $Z = 15$ and $Z = 10$. It is important to mention that although the prediction error is higher there is not a significant propagation of the error between time steps like for the traditional LSTM as a critical limitation described in [27]. For instance, compared to the design proposed in [28] our model has a significantly lower gradient of the propagation of error, i.e., 18 times lower RMSE variation over the normalized data between $L = 10$ and $L = 2$. Our finding in this regard can be visually exemplified in the following figure.

Fig. 9 shows some examples of path loss evolution predictions for $L \{5, 10, 16\}$ steps ahead, considering a sequence history of $Z = 20$ from the dataset of 100 samples. All the samples correspond to the testing subset (i.e., not used for the training and therefore not directly learned by the BiLSTM neural network).

As depicted in Fig. 9a) the error is insignificant for $L = 5$, and increases for $L = 10$, but without a significant propagation of the error across each time step. In Fig. 9c) it can be noticed that the model is even capable of limiting the accumulative propagation of the error and self-correcting. This is thanks to the hybrid prediction mechanism of the BiLSTM neural network architecture, which allows to self-correct the multivariate model after an erroneous prediction at a certain time stamp. A higher RMSE caused by a set of errors in the forward LSTM will not match the backward LSTM cells. Therefore, a large deviation from the model in the forward layer will have a proportional compensation by the backward layer in the next time stamps. For instance, after a model deviation in the prediction of $\beta'$ for the time stamp 9 (in Fig. 9c) the error significantly decreases in the subsequent predictions (i.e., time stamps 12 to 18 in Fig. 9c).

**E. NEURAL NETWORK HYPERPARAMETERS AND COMPUTATIONAL PERFORMANCE**

Fig. 10 shows the inference time for the path loss prediction as the function of the number of BiLSTM cells for different values of $Z$. The best trade-off between accuracy and com-
putational performance is achieved for 256 BiLSTM cells. For this case, the difference in computational performance as a function of $Z$ is lower than 16%. Even for the higher number of BiLSTM cells for which still a good accuracy is achieved (i.e., 512), the performance difference for different sequence lengths does not follow a quadratic dependence, as is the case of the transformer model presented in [27].

The hyper-parameter with the highest impact in the training time is the total number of iterations required for maximizing the neural network accuracy. For $Z = 20$, at least 300 iterations are required for the training phase with a training time of 45s @5 GFLOPS. For $Z = 10$, at least 200 iterations are required with a training time of 24s @5 GFLOPS. As for our small indoor scenario, it was possible to generate a digital twin model, the training time was significantly small. Nevertheless, for online training higher performance hardware might be required to reduce the initial pre-training below 5 seconds.

IV. CONCLUSIONS

In this paper, we have investigated a BiLSTM-based neural network for predicting the path loss variation in a mobile collaborative multi-APs distributed massive MIMO environment. Our proposed algorithm jointly solves two gaps identified in the literature: 1) Abstracting from the spatial domain data and related trajectory, and 2) mitigating the propagation of the error prediction faced by other sequential prediction algorithms.

In terms of accuracy, for this particular dataset, our model is capable of predicting up to 12 time-stamps ahead with an error lower than 0.35 ($95^{th}$-percentile) for a sequence length of only 20 channel samples, and without any information from the spatial domain. Our BiLSTM hybrid prediction is also capable of self-correcting after a burst of errors in a multivariate prediction on the evaluated scenario. Compared to the state-of-the-art the propagation of the RMSE for a multivariate sequential prediction is reduced by a factor of 18 on the test subset.

Future work will consist of evaluating a multi-model approach based on the Digital Cousin paradigm, where multiple small imperfect models are trained and dynamically selected depending on their real-time performance. For different ranges of speeds, we will generate a neural network model and evaluate a dynamic selection of them depending on the accuracy of each model for the real scenario.

REFERENCES


RODNEY MARTINEZ ALONSO (Senior Member, IEEE) obtained a B.Sc. degree in Telecommunications and Electronics Engineering (2010) and a M.Sc. degree in Digital Systems (2015) from Havana University of Technology (CUAJE). He worked as a Research Engineer at LACETEL, Research and Development Telecommunications Institute between 2010 and 2021. In 2016 he joined WAVES group (Department of Information Technology – INTEC, Ghent University) where his main research was focused on dynamic spectrum access technologies. This work led to a PhD degree in Electrical Engineering from Ghent University in June 2020. Currently, he is a senior researcher at KU Leuven, with his main research interest in Artificial Intelligence applications for Next Generation wireless networks.

ROBBERT BEERTEN (Student Member, IEEE) obtained his B.Sc in Engineering and his M.Sc in Electrical Engineering from KU Leuven in 2019 and 2021, respectively. He is currently working as a PhD researcher at the Networked Systems group in the Department of Electrical Engineering, KU Leuven. His research interests include the optimization of wireless networks and distributed signal processing in the context of next-generation networks.

ACHIEL COLPAERT (Student Member, IEEE) obtained his BSc and MSc degrees in electrical engineering from KU Leuven, Belgium respectively in 2015 and 2017. He also obtained a Ph.D. at KU Leuven, focusing on high-throughput wireless links for UAV applications.

ANDREA P. GUEVARA (Member, IEEE) obtained her B.Sc in Electronics and Telecommunications at the University of Cuenca, Ecuador in 2013. In 2015 she got her MSc in Telecommunications and the prize for the best academic performance at Research at King’s College London, UK. In 2022 she got her Ph.D. degree at KU Leuven, her main interests are signal processing, interference analysis of cooperative and distributed massive MIMO systems.

SOFIE POLLIN (Senior Member, IEEE) is a professor at KU Leuven focusing on wireless communication systems. Before that, she worked at imec and UC Berkeley, and she is currently still a principal member of the technical staff at imec. Her research centres around wireless networks that require networks that are ever more dense, heterogeneous, battery-powered, and spectrum constrained. Her research interests are cell-free networks, integrated communication and sensing, and non-terrestrial networks.