Sequence Based Deep Neural Networks for Channel Estimation in Vehicular Communication Systems

SA Ngorima[1,2,3][0000-0002-0775-3529], ASJ Helberg[1][0000-0001-6833-5163], and MH Davel[1,2,3][0000-0003-3103-5858]

[1] Faculty of Engineering, North-West University, South Africa
[2] Centre for Artificial Intelligence Research, South Africa
[3] National Institute for Theoretical and Computational Sciences, South Africa
aldringorima@gmail.com
albert.helberg@nwu.ac.za

Abstract. Channel estimation is a critical component of vehicular communications systems, especially in high-mobility scenarios. The IEEE 802.11p standard uses preamble-based channel estimation, which is not sufficient in these situations. Recent work has proposed using deep neural networks for channel estimation in IEEE 802.11p. While these methods improved on earlier baselines they still can perform poorly, especially in very high mobility scenarios. This study proposes a novel approach that uses two independent LSTM cells in parallel and averages their outputs to update cell states. The proposed approach improves normalised mean square error, surpassing existing deep learning approaches in very high mobility scenarios.

Keywords: Channel estimation · deep learning · dual-cell LSTM · IEEE 802.11p · vehicular channels.

1 Introduction

The international wireless communication standard IEEE 802.11p was introduced for vehicle-to-vehicle communication. However, the original IEEE 802.11p framework did not take into account high mobility and rapid channel changes, which can affect system performance at high speeds [15]. Reliable wireless vehicular communication is challenging due to the need for accurate channel state information (CSI) in high-mobility scenarios, where estimates can quickly become outdated.

IEEE 802.11p employs a data pilot-aided (DPA) approach for channel estimation, with techniques such as spectral temporal averaging (STA) [3] and time-domain reliable test frequency domain interpolation (TRFI) [11] estimators to improve accuracy. However, in high signal-to-noise ratios (SNRs), STA may not effectively account for frequency and time variability while TRFI's assumption of high correlation between successive transmitted signals may not hold true.

The ability of a deep neural network (DNN) to generalize in a robust manner has made it an attractive option to improve channel estimation. DNN algorithms have been investigated as a means of improving traditional channel estimation techniques such as STA and TRFI. These DNN-based postprocessors have excelled at capturing time-varying behaviours in complex propagation environments. An auto-encoder (AE) DNN successfully handled the inherent error propagation challenges in DPA estimation [8]. Another approach combined STA for initial linear channel estimation with a deep neural network (DNN) as a non-linear postprocessor [5]. In Gizzini et al. [6] a TRFI-DNN was introduced that combines TRFI and a DNN. TRFI-DNN has been shown to be more effective than other DNN methods compared with.

Performance-wise, AE-DNN [8] has a relatively low bit error rate (BER). However, it is computationally complex and does not effectively learn the time-frequency channel correlation. STA-DNN [5] outperforms both AE-DNN and TRFI-DNN [6] in low SNRs of up to 20 dB at a velocity of 120 km/h. However, its performance starts to deteriorate as the SNR increases. TRFI-DNN, on the other hand, outperforms AE-DNN by approximately 3 dB for BER $= 10^-3$. In scenarios where mobility is higher than 120 km/h, STA-DNN starts to experience an error floor at even lower SNRs of 17 dB. However, the challenge of dealing with a rapidly changing channel remains for both temporal filtering and DNN postprocessing.

To estimate wireless channels in high-mobility situations, we propose using a dual-cell long-short-term memory (LSTM) network. This network can capture short-term temporal dependencies and causal correlations. A DPA block and a temporal averaging (TA) block are used to further reduce the noise in the dual-cell LSTM estimates.

The remainder of the paper is structured as follows: Section 2 reviews IEEE 802.11p and channel estimation, Section 3 introduces the dual-cell LSTM approach, Section 4 describes the experiments, Section 5 presents results and analysis, and Section 6 concludes.

2 Background

This section provides a brief overview of the IEEE 802.11p standard specification. It also discusses channel estimation system models and techniques within the context of IEEE 802.11p, serving as the foundation for improving vehicular communication systems.

2.1 IEEE 802.11p Standard Specifications and Frame Structure

The IEEE 802.11p protocol employs a technique referred to as Orthogonal Frequency Division Multiplexing (OFDM) to transfer data via radio channels. OFDM divides the available bandwidth into several closely spaced subcarrier frequencies, allowing many signals to be sent simultaneously. During transmission, some subcarriers carry pilot signals that are known by both the transmitter and the receiver. The receivers use pilots to analyse changes in the communication channel. As vehicles travel at different speeds, factors such as distance, barriers, and multipath effects constantly change the way signals propagate. These pilot signals can be used to track time-varying channel conditions and obtain the CSI. This CSI can then be used to improve the decoding of transmitted user data, ensuring efficient communication.

OFDM signal consists of modulation symbols sent on an active subcarrier during a particular time interval. OFDM frames are made up of several payload OFDM signals that are preceded by preamble symbols that fulfil various key roles. The preamble enables the receiver to synchronise, establish an initial coarse channel estimate, and mark the beginning of incoming frames.

In IEEE 802.11p, only 52 of the 64 available subcarriers are used for data transmission and pilot symbols, as illustrated in Figure 1. The rest of the subcarriers are kept for other purposes, such as guard bands and direct current (DC) offset. To ensure accurate channel tracking, pilot signals are inserted into specific subcarriers.

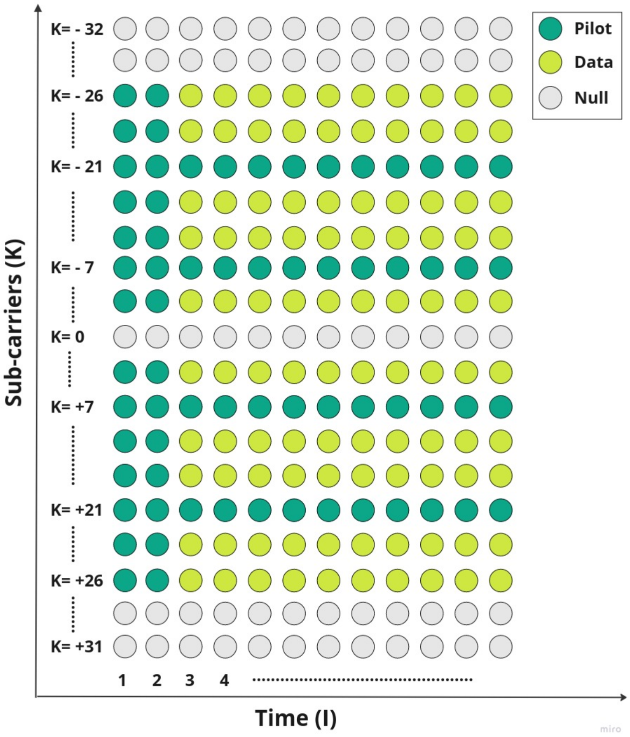


Fig. 1: Subcarrier arrangements in a IEEE802.11p transmission [5].

2.2 Channel Estimation System Model

This study assumes perfect synchronisation and only considers a frame structure comprising two extended preambles at the start, followed by I OFDM data symbols. Following the terminology used in [4], the expression of the transmitted and received OFDM symbol can be represented as follows:

$$\tilde{y}_i[k] = \begin{cases} \tilde{h}_{i,d}[k]\tilde{x}_{i,d}[k] + \tilde{v}_{i,d}[k], & k \in \mathcal{K}_{\mathrm{d}} \\ \tilde{h}_{i,p}[k]\tilde{x}_{i,p}[k] + \tilde{v}_{i,p}[k], & k \in \mathcal{K}_{\mathrm{p}} \end{cases}, \tag{1}$$

where $\tilde{x}_i[k]$ denotes the OFDM symbol transmitted at the i^{th} instance, affected by its corresponding time-varying frequency-domain channel response $\tilde{h}_i[k]$. $\tilde{y}_i[k]$ are the OFDM data symbols received. Furthermore, $\tilde{v}_i[k]$ denotes the frequency-domain equivalent of additive white Gaussian noise (AWGN) characterised by a variance of σ^2. The indices K_d and K_p refer to the sets of subcarriers dedicated to data and pilot signals, respectively.

Thus, the OFDM symbol received ($\tilde{y}_i[k]$) consists of both the data and the pilot subcarrier symbols affected by the channel response, with noise added. Channel estimation is the task of determining $\tilde{h}_i[k]$ for each OFDM symbol.

2.3 IEEE 802.11p Channel Estimation Schemes

Accurate channel estimation in OFDM systems such as IEEE 802.11p relies on well-allocated pilot subcarriers. However, the standard's limited number of pilots may not suffice for tracking channel variations effectively. Two proposed channel estimation schemes tackle this issue: one uses data subcarriers alongside pilots, while the other inserts additional pilots at the cost of a reduced transmission rate. Achieving reliable channel estimation requires a careful balance between accuracy and transmission efficiency.

Recently, models based on recurrent neural networks (RNNs) have shown promise in leveraging sequential dependencies in channel data for channel estimation in vehicular communication [4, 10, 13]. Among RNN architectures, LSTM networks can learn long-term dependencies and have shown excellent results in sequential tasks [9].

In the context of channel estimation, an LSTM has been used to directly map pilot sequences to channel responses [13], while Gizzini et al. proposed a long-short-term memory data pilot-aided temporal averaging (LSTM-DPA-TA) integration to improve estimates in high mobility situations [4]. Hou et al. [10] also employed a single-layer Gated Recurrent Unit (GRU) as a post-processing for the estimation of DPA.

DPA Estimation The DPA estimation technique uses both pilot subcarriers and demapped data subcarriers to estimate the channel for the present OFDM symbol [8]. Following the notation in [4] the demapped data symbol $d_i[k]$ for each subcarrier $\mathcal{K}$ is given in Equation 2 as follows:

$$d_i[k] = \mathfrak{D}\left(\frac{y_i[k]}{\hat{\tilde{h}}_{\mathrm{DPA}_{i-1}}[k]}\right), \hat{\tilde{h}}_{\mathrm{DPA}_0}[k] = \hat{\tilde{h}}_{\mathrm{LS}}[k], k \in \mathcal{K}_{\mathrm{on}}, \tag{2}$$

Here, $\mathfrak{D}(.)$ signifies the process of demapping to the closest constellation point based on the chosen modulation order, $\mathcal{K}_{\mathrm{on}}$ are the active subcarriers, $\tilde{h}_{\mathrm{LS}}[k]$ is the channel estimated using the Least Squares (LS) method from the received preambles, namely $y_1^{(p)}[k]$ and $y_2^{(p)}[k]$ such that:

$$\tilde{h}_{\mathrm{LS}}[k] = \frac{y_1^{(p)}[k] + y_2^{(p)}[k]}{2p[k]}, k \in \mathcal{K}_{\mathrm{on}}, \tag{3}$$

In this context, $p[k]$ denotes the predetermined frequency-domain preamble sequence. Subsequently, the final updates for the DPA channel estimation are carried out in the following manner:

$$\tilde{h}_{\mathrm{DPA}_i}[k] = \frac{y_i[k]}{d_i[k]}, k \in \mathcal{K}_{\mathrm{on}}. \tag{4}$$

It should be noted that DPA fundamental estimation serves as an initial reference for a majority of IEEE 802.11p channel estimation techniques [10].

TA Processing Temporal averaging (TA) is a noise reduction technique in which the noise reduction ratio can be calculated analytically [4]. The TA method assumes that the noise terms of consecutive OFDM symbols are not correlated. TA analyses variations by averaging the channel estimations across OFDM symbols. Following the notation in [4], consider the estimate of the channel $h_{k,n}$ in the subcarrier k and symbol n. TA computes a moving estimate $\hat{H}_k$ of the channel frequency response h_k as the weighted sum:

$$\hat{H}_{k,n} = (1 - 1/\alpha)\hat{h}_{k-1,n} + (1/\alpha)\hat{h}_{k,n} \tag{5}$$

In this case, α denotes the window size over which the averages are calculated. A larger α incorporates more symbols. TA reduces noise through averaging, making it a good candidate for postprocessing.

STA-DNN Estimator Previous work [5] recognised the problems in only using traditional STA estimates [3] and proposed a hybrid STA-DNN technique to solve them. The typical STA estimator averages the initial estimate of the DPA channel over frequency and time.

To compensate for the time-varying channel conditions, STA uses fixed average window widths and weights [3]. To counteract this, Gizzini et al. [5] suggest creating an STA-DNN estimator by feeding the STA estimate to a deep neural network (DNN). Their evaluation showed that the STA-DNN technique corrects errors and greatly increases the accuracy of the estimation over the traditional STA estimation [5]. Despite these improvements, there is still a significant error in high-mobility vehicle situations at low SNR [4].

LS Channel Estimate In IEEE 802.11p, the basic LS channel estimation technique is used as initial channel estimation. Two successive training symbols T_1 and T_2 are used to obtain the initial frequency response of the channel of the k^{th} subcarrier as follows:

$$\tilde{H}_0\left(k\right) = \frac{Y_{T1}\left(k\right) + Y_{T2}\left(k\right)}{2X_T\left(k\right)}, k \in \mathcal{K}_{\mathrm{on}} \tag{6}$$

where $X_T(k)$ is the predefined training symbol of the k^{th} subcarrier, $Y_{T1}(k)$ and $Y_{T2}(k)$ are the symbols in the frequency domain of the receiver. The LS channel estimation approach overlooks time variations, resulting in reduced accuracy as the OFDM symbol index increases. To maintain reliable performance in mobile scenarios, more advanced channel tracking techniques are essential, accounting for dynamic channel changes and ensuring accurate estimation.

The channel estimation schemes discussed are used in this work for comparison purposes. LS estimation is used as an initial estimation approach before our dual-cell LSTM method. This is followed by applying DPA as a post-processing step to refine our dual-cell LSTM estimate. A step-by-step procedure on how these techniques are used in our work is discussed in detail in Section 3.

3 Proposed Dual-Cell LSTM Estimation Method

LSTM networks are designed for sequential data with temporal dependencies. These networks have an architecture that allows them to understand the temporal dependencies in the data, enabling them to predict future data based on past observations [9]. The gated cell structure of LSTMs enables them to learn long-term temporal relations by regulating the input of new information, eliminating unnecessary past information, and updating the hidden state using selected cell state values [14]. LSTMs have been successfully used in different domains and applications such as time series forecasting [12] and speech recognition [7].

This section introduces a novel dual-cell LSTM architecture that uses two independent LSTM cells in parallel to capture sequential information from different temporal perspectives. This enables the model to fuse representations learnt from different, non-overlapping temporal perspectives at each timestep. The method also integrates DPA and TA as post-processing methods to reduce noise.

The dual-cell structure is distinct from a bidirectional LSTM as it is composed of two parallel LSTMs operating in the forward direction, while a bidirectional LSTM comprises two LSTMs, one processing the sequence in a forward direction and the other in a backward direction. Our hypothesis is that this dual-cell architecture is capable of capturing a more complete image of channel dynamics compared to other LSTM structures. The proposed protocol aims to efficiently address channel estimation in highly dynamic environments, such as vehicular communication.

As illustrated in Figure 2, the inputs (x_{t-1}, x_{t-2}, x_{t-3}, etc.) are supplied directly to the independent LSTM cells at the same time. Following the typical

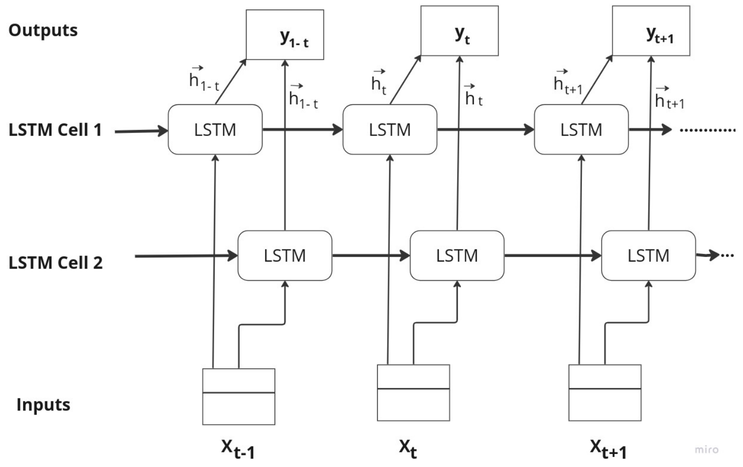


Fig. 2: Schematic representation of the dual-cell LSTM estimator. The architecture consists of two parallel LSTM cell chains, with each cell processing sequential inputs from x_{t-1} to x_t. The outputs from both chains are fused to produce the final output.

LSTM design [9], each LSTM cell consists of an input gate, a forget gate, an output gate, and a cell memory block. For both cells, the number of hidden units is set to H. At each time step t, the same input sequence x_t is fed into both cells to update their hidden states ($h_{1,t}$ and $h_{2,t}$) and cell states ($c_{1,t}$ and $c_{2,t}$). This processing takes place in parallel, without information exchange between cells. Finally, a combined output y_t is produced by averaging the hidden state outputs of both cells and passing them through a linear layer. This captures representations from the two processing streams, although independently and simultaneously. Averaging the outputs of the two cells has various benefits:

- Combining the representations learned by each cell helps to prevent overfitting. Cells may capture somewhat different features of the input sequence.
- Averaging provides a type of ensembling in which the combination of numerous models (cells) outperforms isolated ones.
- By averaging time-distributed outputs at each step, the model may use information processed in both cells at the same time.

3.1 Dual-Cell LSTM Estimation Algorithm

This section presents the algorithm for the proposed dual-cell LSTM model. (Also see Algorithm 1.)

The Dual-Cell LSTM model for the channel estimation method can be broken down into four phases:

- First, an LS channel estimation is performed. Initial channel estimation is obtained as illustrated in Section 2.3. This forms the input dataset ($\tilde{\bar{x}}_i$) used

Algorithm 1 Dual-Cell LSTM Model for Channel Estimation

```
Data: HLS_Structure, True_Channel_Structure,
  Step 1: LS estimation to obtain the initial channel estimation that is used as
  the training dataset (HLS_Structure)
  Step 2: Load dataset
    -   Load True_Channel_Structure
    -   Load HLS_Structure
  Step 3: Prepare input and target matrices
    -   Construct Dataset_X by combining real and imaginary parts of
  HLS_Structure.
    -   Construct Dataset_Y by combining real and imaginary parts of
  True_Channel_Structure.
  Step 4: Initialize dual LSTM cell model
    -   Create two LSTM cells: cell1 and cell2
    -   Initialise hidden and cell states of both cells
  Step 5: Train model
  while not converged do
      Step 5.1: Forward pass
      -     Feed Dataset_X to both cells
      -     Get outputs out1 and out2 from cell1 and cell2
      -     Calculate out_avg = (out1 + out2)/2
      -     Calculate loss between out_avg and Dataset_Y
      Step 5.2: Backpropagate loss
      -     Update cell parameters using backpropagation
  end while
  Step 6: Channel estimation on new data
    -   Forward pass new samples through trained model
    -   Get channel estimates $\hat{est} = out\_avg$
  Step 7: Post-processing
    -   Perform DPA estimation on $\hat{est}$ to improve estimates
    -   Perform TA on DPA outputs
    -   Final estimated channel
  Step 8: Calculate evaluation metrics
    -   Calculate NMSE, BER., on test dataset
  Step 9: Repeat step 4 for the required number of epochs
  Step 10: Save trained model
```

to train the model. Following the notation presented in [4], the input $\bar{x}_i$ can be obtained as follows:

$$\bar{x}_i = \begin{cases} \hat{\hat{h}}_{\mathrm{LSTM}_{i-1,d}}[k], & k \in \mathcal{K}_{\mathrm{d}} \\ \hat{\hat{h}}_{i-1,p}[k], & k \in \mathcal{K}_{\mathrm{p}} \end{cases}. \tag{7}$$

The input $\tilde{\bar{x}}_i$ is computed by transforming LS estimate $\bar{x}_i$ from complex values to real values. $\hat{\hat{h}}_{i-1,p}[k]$ is the LS estimated channel at the K_p subcarriers. $\tilde{\bar{x}}_i$ is input to the LSTM as:

$$\hat{\hat{h}}_{\mathrm{LSTM}_{i.d}} = \mathit{\Omega}_{\mathrm{LSTM}}(\tilde{\bar{x}}_i,\ \mathit{\Theta}), \tag{8}$$

where $\mathit{\Omega}_{\mathrm{LSTM}}$ denotes the LSTM processing unit and $\mathit{\Theta}$ denotes the overall weights.

- Second, the LSTM model is trained to estimate the channel characteristics from the received data.
- Third, the DPA post-processing technique is used to improve estimations, as follows:

$$d_{\mathrm{LSTM}_i}[k] = \mathfrak{D}\left(\frac{y_i[k]}{\hat{\hat{h}}_{\mathrm{LSTM}_{i-1}}[k]}\right), \hat{\hat{h}}_{\mathrm{LSTM}_0}[k] = \hat{\hat{h}}_{\mathrm{LS}}[k], \tag{9}$$

$$\hat{\hat{h}}_{\mathrm{LSTM-DPA}_i}[k] = \frac{y_i[k]}{d_{\mathrm{LSTM}_i}[k]}. \tag{10}$$

- Fourth, the TA technique is the final post-processing method: TA is applied to the estimated channel $\hat{\hat{h}}_{\mathrm{LSTM-DPA}_i}[k]$ to further reduce the impact of AWGN noise as follows:

$$\hat{\hat{h}}_{\mathrm{DNN-TA}_{i,d}} = \left(1 - \frac{1}{\alpha}\right)\hat{\hat{h}}_{\mathrm{DNN-TA}_{i-1,d}} + \frac{1}{\alpha}\hat{\hat{h}}_{\mathrm{LSTM-DPA}_{i,d}}. \tag{11}$$

  A fixed α value of 2 is used this is based on the analysis done in [4].

Finally, estimates are evaluated using normalised mean squared error (NMSE) and bit error rate (BER) metrics.

4 Experimental Setup

This section discusses the channel model employed for data generation, the data preparation for LSTM and the hyperparameter optimisation process.

4.1 Channel Model

Vehicular channel models, widely examined in the literature [4–6], originate from channel measurements in metropolitan Atlanta, Georgia, USA [1]. We conducted an analysis of the vehicle-to-vehicle same direction with wall (VTV-SDWW) tapped delay line (TDL) vehicular channel model, which is used for communication between two vehicles travelling in the same direction with a central wall between them and maintaining a distance of 300-400 metres between them.

Two mobility scenarios based on the VTV-SDWW TDL were adopted for comparison: (i) high mobility (V = 100 km/h, Doppler shift f_d = 550 Hz) and (ii) very high mobility (V = 200 km/h, f_d = 1,100 Hz). Simulation parameters included a frame size of 50 OFDM symbols, 16QAM modulation, and convolutional channel coding at a half-code rate. The dataset consisted of 12,000 training samples, 4,000 validation samples, and 2,000 testing samples. During training, a training SNR level of 40 dB is used to improve the generalization of the model. Each simulation generated a packet of 50 OFDM symbols sent across a simulated wireless channel, capturing varying channel conditions. In this study, a 'packet sample' represents one 50-OFDM symbol packet, serving as a single observation for estimation.

4.2 Data Preparation for LSTM

The dataset that was created according to the IEEE 802.11p standard (see Section 2). We excluded the signal field from the dataset, assuming optimal receiver synchronisation, for the sake of simplicity. As a result, the emphasis is on the presence of two extended training symbols within each transmitted frame, followed by a set of I Orthogonal Frequency Division Multiplexing (OFDM) data symbols. The received OFDM symbol is represented by Equation 1.

We constructed $Dataset\_X$ and $Dataset\_Y$ matrices for the input and target data, respectively. For training data, we obtained a real-valued training dataset $Dataset\_X$ by concatenating the real and imaginary parts of the propagating channel ($HLS\_Structure$) into one vector. $Dataset\_Y$ is the target output matrix created similarly to $Dataset\_X$ by concatenating the real and imaginary parts of the actual or ground truth channel for specific positions, 1 to 48 for real and 49 to 96 for imaginary.

$Dataset\_X$ has dimensions $12,000 \times 50 \times 104$. Similarly, $Dataset\_Y$ has dimensions $12,000 \times 50 \times 96$. The first dimension is the size of the dataset. The second dimension denotes the number of OFDM symbols per frame, while the third is the sequence length, which is the size of the input data positions if it is $Dataset\_X$ or output if its $Dataset\_Y$. Our input size is 104 and our output size is 96 (as above).

4.3 Hyperparameter Optimisation

The hyperparameters of the dual-cell LSTM model were optimised using Optuna to find the values that minimise loss of validation during training. Optuna is a Bayesian optimisation framework that generates hyperparameter configurations using a tree-structured parzen estimator (TPE) to maintain a probabilistic model that links hyperparameters to measurable outcomes [2]. It terminates underperforming tests early to maximise search efficiency. The procedure is repeated until either the desired convergence is achieved or the maximum number of trials is reached.

We optimised a set of hyperparameters, namely the learning rate, the step size of the optimiser, the gamma of the *StepLR* scheduler, the batch size, the weight decay and the dropout probability. The optimisation procedure was carried out as follows:

1. A dual LSTM model with fixed input size (104) was defined. This is the total number of input subcarriers used in this work ($K_{on} * 2$ where $K_{on} = 52$ active subcarriers). The LSTM size was 128.
2. The non-dominated Sorting Genetic Algorithm II (NSGA-II) sampler created an Optuna study for multi-objective optimisation.
3. An objective function trained the dual-cell LSTM for 100 epochs with specified hyperparameters (Table 1).
4. Optuna recommended hyperparameters within defined limits, and Adam optimiser initialised and trained the dual-cell LSTM.

5. Training and validation losses were tracked for each epoch, with the average validation loss used.
6. The study ran 150 optimisation trials to identify optimal hyperparameters based on the lowest validation loss.
7. The loss curve, weight, and bias plots were logged to the weights and bias[4] for visualisation.
8. Optimal trial hyperparameters (learning rate, step size, gamma, dropout, weight decay) were determined on the basis of the lowest validation loss.

The final model is trained for 250 epochs using a batch of 128. A summary of the parameters of the dual cell LSTM model used in this work is given in Table 1.

Table 1: Optuna Hyperparameter optimisation: 150 trials and 100 epochs

Hyperparameters	**Search space**	**Final value**
Learning rate	[1e-5, 1e-1]	0.01
Step size	[1, 10]	9
gamma	[0.1, 1]	0.6
Dropout probability	[0.0, 0.8]	0.002
Weight decay	[1e-6, 1e-2]	0.008
Batch size	[16, 32, 48, . . . ,128]	128

5 Analysis and Simulation Results

We evaluate the following channel estimation methods, STA-DNN [5], LSTM-DNN-DPA [13], LSTM-DPA-TA [4], DPA-TA and the proposed dual-cell LSTM estimator against a perfect channel by calculating BER and NMSE in MATLAB. The hidden layer dimensions of the STA-DNN were 15-15-15 and 128-40 for the LSTM-DNN.

1. **LSTM-DNN-DPA:** This method involves cascading an LSTM and a multilayer perceptron network (MLP) to estimate the channel and DPA as a post processor [13]. However, overfitting was not evaluated in this work, as cascaded models possess a high risk of overfitting due to an increased number of parameters.
2. **LSTM-DPA-TA (128):** This approach uses an LSTM model for channel tracking followed by DPA and TA for noise reduction. Overall performance is based on the precision of the initial LSTM estimate. The noise mitigation ratio of TA processing has been analytically derived in [4]. This method was not evaluated against other LSTM architectures.

[4] https://wandb.ai/

3. **LSTM-DPA-TA (64):** This is the same LSTM-based method as above but using a smaller LSTM: size 64 instead of 128 [4]. A smaller LSTM size struggles to learn long-term dependencies in dynamic environments compared to a larger LSTM.
4. **STA-DNN:** This technique utilises DNNs to capture additional temporal and frequency correlations [5]. However, this method does not consider sequential data, thus limiting its capability.
5. **DPA-TA:** We established our lower bound by using only the DPA and TA techniques. The shortcoming of this approach is that it only uses conventional techniques which are limited to learning patterns in data, especially dynamic channels.
6. **Dual-cell LSTM:** We evaluate the performance of our dual-cell LSTM network approach against the other methods in the same scenarios discussed below.

5.1 High Mobility Scenario

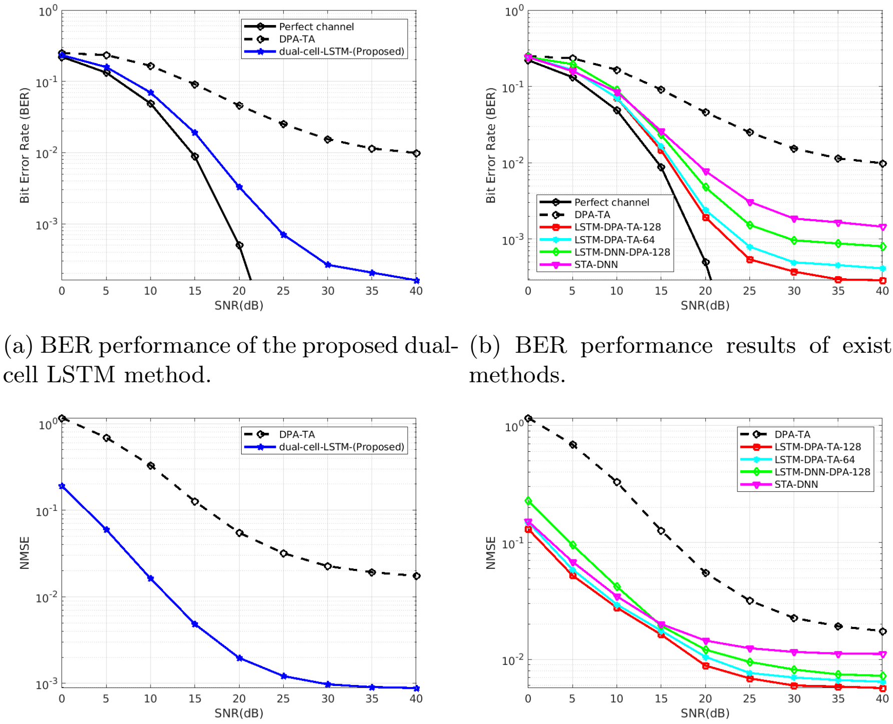
In this section, we evaluate the performance of existing approaches and our proposed dual-cell LSTM method under high mobility conditions, characterised by a velocity (v) of 100 km/h and a Doppler frequency (f_d) of 550Hz using two key metrics; BER and NMSE.

Clarification on Data Comparison: The results of existing methods have been extracted directly from Gizzen et al. [4], while the dataset used to develop the dual-cell LSTM method was generated in-house following the experimental setup detailed in that respective paper. The performance of the proposed method is compared against the stated published performance of the methods presented in [4]. We caution that in this comparison, the datasets generated according to Section 4.2 and the unpublished dataset in [4] may differ, although the described vehicular scenarios were precisely duplicated. To reflect this caution, we present the comparison in separate graphs in Figures 3 and 4.

Figures 3a and 3b show the BER performance results of the DNN-based estimators and the classical DPA-TA estimator. Our experiments showed that the dual-cell LSTM was capable of dealing with a variety of SNR levels. At SNR levels between 0 and 12 dB, it performed similarly to LSTM-DPA-TA (128). When the SNR was between 27 and 40 dB, it outperformed all the models tested. Figures 3c and 3d compare the dual-cell LSTM NMSE performance with that of existing techniques. Lower NMSE values indicate a better estimate of the true channel response. At an NMSE of 10^{-2}, the dual-cell LSTM method shows better performance than LSTM-DA-TA (128), LSTM-DA-TA (64) and LSTM-DNN by approximately 7 dB, 8 dB, and 12 dB improvements, respectively. At 35 dB SNR, it reaches an NMSE of 10^-3.

5.2 Very High Mobility Scenario

This section evaluates channel estimation methods under very high mobility conditions of 200km/h and a frequency Doppler of 1,100Hz. BER and NMSE



(a) BER performance of the proposed dual-cell LSTM method.

(b) BER performance results of existing methods.

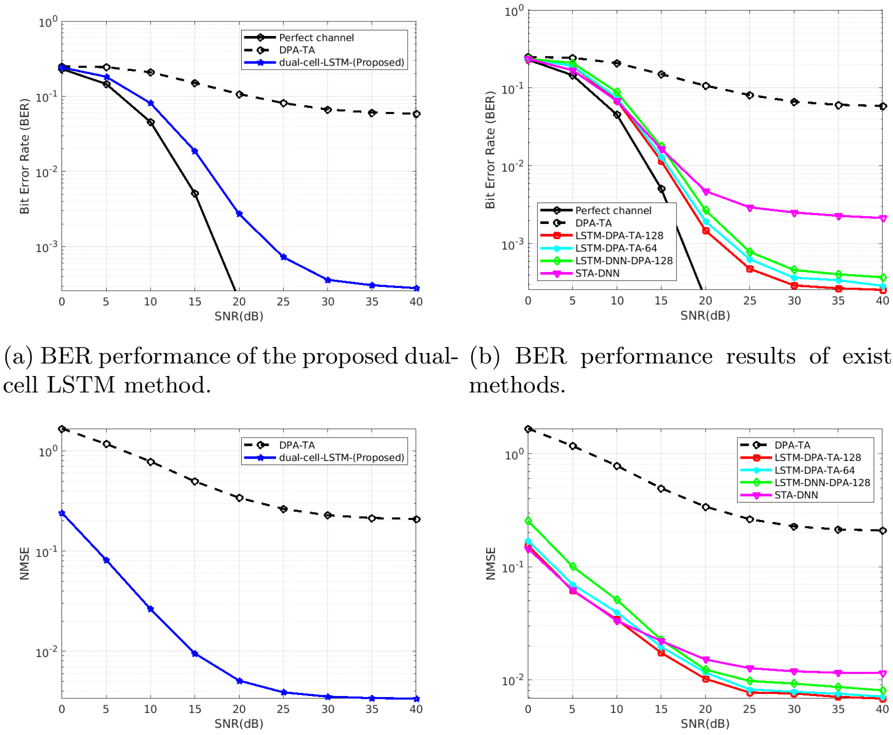
(c) NMSE performance of the proposed dual-cell LSTM method.

(d) NMSE performance of existing methods.

Fig. 3: BER and NMSE results under High Mobility conditions using the VTV-SDWW channel model at $v = 100$ km/h, $f_d = 550$ Hz.

metrics are used to evaluate performance. We again follow the experimental setup of Grizzini et al. [4] to simulate the very high mobility environment, and present results separately. Figures 4a and 4b plot the BER against SNR, where all estimators showed reduced performance due to fast fading that causes a decrease in the temporal correlations of the channel.

At SNR levels ranging from 0 to 15 dB, the LSTM-DPA-TA (128) approach surpasses the dual-cell LSTM technique by up to 0.5 dB. On the other hand, the BER of feedforward models such as STA-DNN deteriorates significantly more rapidly even at high SNR due to its inability to exploit temporal dependencies in highly dynamic channels. The corresponding NMSE vs SNR graphs are shown in Figures 4c and 4d. Similarly, despite the challenges of extremely high mobility, dual-cell LSTM showed a capability to achieve a very low NMSE by a significant margin compared to the other existing DNN methods.



(a) BER performance of the proposed dual-cell LSTM method.

(b) BER performance results of existing methods.

(c) NMSE performance of the proposed dual-cell LSTM method.

(d) NMSE performance of existing methods.

Fig. 4: BER and NMSE performance at Very High Mobility: VTV-SDWW channel model at V = 200km/h, f_d = 1,100Hz.

6 Conclusion

This paper introduced a dual-cell LSTM network model for dynamic channel estimation in vehicular communication systems. A comparison was conducted with results available from the literature, with experimental setups duplicated. Our dual-cell LSTM model showed the potential to achieve very low NMSE between estimated and true channel responses across all tested SNR levels of magnitude close to 1 compared to other sequence models. In terms of BER, the dual LSTM method showed improved performance in high SNRs, starting at 30 dB, performing better than the other methods. This result supports our hypothesis that the combination of two LSTM methods in a parallel ensemble can outperform the existing single LSTM method and the feedforward methods. Future work will include evaluating the complexity and learning capacity of ensemble sequential learning methods.

Acknowledgements The authors are grateful to NITheCS, the Telkom CoE at the NWU and Hensoldt South Africa, who supported this research.

References

1. Acosta-Marum, G., Ingram, M.A.: Six time-and frequency-selective empirical channel models for vehicular wireless lans. IEEE Vehicular Technology Magazine **2**(4), 4–11 (2007)
2. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2623–2631 (2019)
3. Fernandez, J.A., Borries, K., Cheng, L., Kumar, B.V., Stancil, D.D., Bai, F.: Performance of the 802.11 p physical layer in vehicle-to-vehicle environments. IEEE transactions on vehicular technology **61**(1), 3–14 (2011)
4. Gizzini, A.K., Chafii, M., Ehsanfar, S., Shubair, R.M.: Temporal averaging lstm-based channel estimation scheme for ieee 802.11 p standard. In: 2021 IEEE Global Communications Conference (GLOBECOM). pp. 01–07. IEEE (2021)
5. Gizzini, A.K., Chafii, M., Nimr, A., Fettweis, G.: Deep learning based channel estimation schemes for ieee 802.11 p standard. IEEE Access **8**, 113751–113765 (2020)
6. Gizzini, A.K., Chafii, M., Nimr, A., Fettweis, G.: Joint trfi and deep learning for vehicular channel estimation. In: 2020 IEEE Globecom Workshops (GC Wkshps. pp. 1–6. IEEE (2020)
7. Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing. pp. 6645–6649. Ieee (2013)
8. Han, S., Oh, Y., Song, C.: A deep learning based channel estimation scheme for ieee 802.11 p systems. In: ICC 2019-2019 IEEE International Conference on Communications (ICC). pp. 1–6. IEEE (2019)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
10. Hou, J., Liu, H., Zhang, Y., Wang, W., Wang, J.: Gru-based deep learning channel estimation scheme for the ieee 802.11p standard. IEEE Wireless Communications Letters **12**(5), 764–768 (2023). https://doi.org/10.1109/LWC.2022.3187110
11. Kim, Y.K., Oh, J.M., Shin, Y.H., Mun, C.: Time and frequency domain channel estimation scheme for ieee 802.11 p. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). pp. 1085–1090. IEEE (2014)
12. Li, Y., Zhu, Z., Kong, D., Han, H., Zhao, Y.: Ea-lstm: Evolutionary attention-based lstm for time series prediction. Knowledge-Based Systems **181**, 104785 (2019)
13. Pan, J., Shan, H., Li, R., Wu, Y., Wu, W., Quek, T.Q.: Channel estimation based on deep learning in vehicle-to-everything environments. IEEE Communications Letters **25**(6), 1891–1895 (2021)
14. Staudemeyer, R.C., Morris, E.R.: Understanding lstm - a tutorial into long short-term memory recurrent neural networks. ArXiv **abs/1909.09586** (2019), https://api.semanticscholar.org/CorpusID:202712597
15. Zhao, Z., Cheng, X., Wen, M., Jiao, B., Wang, C.X.: Channel estimation schemes for ieee 802.11 p standard. IEEE intelligent transportation systems magazine **5**(4), 38–49 (2013)