EDChannel: Channel Prediction of Backscatter Communication Network Based on Encoder-Decoder

Dengao Li
Taiyuan University of Technology
Yongxin Wen
Taiyuan University of Technology
Shuang Xu
Taiyuan University of Technology
Qiang Wang
Taiyuan University of Technology
Ruiqin Bai
Taiyuan University of Technology
Jumin Zhao (zhaojumin@tyut.edu.cn)
Taiyuan University of Technology https://orcid.org/0000-0002-9049-4053

Research Article

Keywords: Backscatter communication, Channel prediction, Deep learning, Encoder-decoder

Posted Date: January 21st, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1173737/v1

EDChannel: Channel prediction of backscatter communication network based on encoder-decoder

Dengao Li[1,3], Yongxin Wen[2,4], Shuang Xu[1,3], Qiang Wang[2,4], Ruiqin Bai[2,4] and Jumin Zhao[2,4*]

[1]Shanxi, College of Data Science, Taiyuan University of Technology, Jinzhong, 030600, China.
[2]Shanxi, College of Information and Computer, Taiyuan University of Technology, Jinzhong, 030600, China.
[3]Technology Research Centre of Spatial Information Network Engineering of Shanxi, Jinzhong, 030600, China.
[4]Intelligent Perception Engineering Technology Centre of Shanxi, Jinzhong, 030600, China.

*Corresponding author(s). E-mail(s): zhaojumin@tyut.edu.cn; Contributing authors: lidengao@tyut.edu.cn; wyx1142557393@163.com; xushuang@tyut.edu.cn; wangqiang510@126.com; bairuiqinty@163.com;

Abstract

Backscatter communication networks have attracted much attention due to their small size and low power waste, but their spectrum resources are very limited and are often affected by link bursts. Channel prediction is a method to effectively utilize the spectrum resources and improve communication quality. Most channel prediction methods have failed to consider both spatial and frequency diversity. Meanwhile, there are still deficiencies in the existing channel detection methods in terms of overhead and hardware dependency. For the above reasons, we design a sequence-to-sequence channel prediction scheme. Our scheme is designed with three modules. The channel prediction module uses an encoder-decoder based deep learning model (EDChannel) to predict the sequence of channel indicator measurements. The channel detection module decides whether to perform a channel detection by a trigger that reflects the prediction effect. The channel selection module performs channel selection based on the channel coefficients of the prediction results. We use a commercial reader to collect data in a real environment, and build an EDChannel model based on the deep learning module of Tensorflow and Keras. As a result, we have implemented the channel prediction module and completed the overall channel selection process. The experimental results show that the EDChannel algorithm has higher prediction accuracy than ARIMA, linear regression (Linear), and autoregression (AR). The overall throughput of our scheme is improved by approximately 2.9% and 17.4% over random frequency hopping in both stable and unstable environments.

Keywords: Backscatter communication, Channel prediction, Deep learning, Encoder-decoder

1 Introduction

Backscatter communication network, a short-distance, ultra-low power waste, low-cost wireless communication technology, is widely used to build large-scale deployment of sensor networks [1, 2]. With the rapid development of the Internet of Things technology and the increasing demand for wireless sensor networks, backscatter communication technology has attracted widespread attention from academic circles and industrial circles, and many potential application scenes have also been mentioned a lot such as target positioning, gesture recognition, and medical testing [3], etc. At the same time, backscatter communication networks have been widely used in various wireless platforms such as Radio Frequency Identification (RFID) [4], Wi-Fi, and Bluetooth, etc. As a result, we can expect that in the near future, devices related to backscatter communication networks will be used on a large scale in everyday life.

However, the backscatter communication network uses the energy obtained by the radio frequency signal at the transmitting end to transmit data [5]. This low-power transmission method makes the quality of the communication not guaranteed in dynamic channels. At the same time, backscatter communication networks are usually deployed in complex network environments, such as factories, shelves and so on. The above-mentioned factors cause the state of the channel to exhibit a high degree of burstiness, which seriously affects the communication quality of the backscatter communication network. Therefore, researchers generally reduce transmission loss in two ways to ensure the higher throughput of the backscatter communication network. One is selecting channels through predicted channel metrics, and selecting channels with better channel quality for communication [6]; the other is adapting rate based on the predicted channel metrics, which dynamically adjusts the communication rate according to the changes in channel quality [7]. Studying the channel prediction in the backscattering communication network is extremely important. Through extensive preliminary experiments, we observe that there are many causes of channel bursts in backscatter communication networks, such as tag collisions due to the addition of new tags, sudden changes in channel metrics due to tag movement, changes in multipath propagation due to the addition of obstacles, or a surge in external interference. Additionally, the inherent dynamic variability and unpredictability of wireless channels also make real-time and accurate channel quality predictions face huge challenges.

Most of the previous research in the field of channel prediction has focused on constructing channel models by designing Markov models [8]. Although this simple model-based approach has specific advantages when computing power and data are limited, their studies generally assume a variety of ideal conditions. And these assumptions are usually difficult to put into actual systems, which leads that such methods are unable to work well in different real environments. In todays data-driven Internet-of-Things era, channel prediction algorithms based on classic machine learning have been applied to the channel prediction of backscatter communications to estimate the channel parameters when the tag is in different states, such as the eigenvalue decomposition (EVD) based on the received signal covariance matrix [9], the expectation maximization (EM) algorithm [10], and the least squares (LS) algorithm [11].

In addition, with the rapid development of cloud computing and big data technologies over the past decade [12], coupled with the continuous growth of the field of machine learning, researchers have begun to try to apply deep learning machine learning which consumes more computing power to channel prediction, so as to obtain more accurate predictions of channel metrics [13, 14]. This provides us with an opportunity to design a more robust and accurate channel prediction system. Specifically, we introduce sequence-to-sequence encoder-decoder models based on deep learning, which are very suitable for the problem of mapping input sequences to output sequences [15]. The model has been widely used in tasks such as text generation and natural language translation. Moreover, the applicability of the encoder-decoder model in predicting channel metrics is also proved in [16].

In this paper, an encoder-decoder based sequence-to-sequence deep learning model (EDChannel) is designed for channel prediction in backscattered communication networks with convolutional neural networks (CNN)+ long short-term memory (LSTM) as the encoder and LSTM as the decoder. We next design a channel detection scheme that has no requirement in hardware dependence based on the sequence-to-sequence prediction mode. The root mean square error (RMSE) of the real-time feedback sequence of the current channel indicator and the predicted value sequence obtained by the prediction module is used to determine whether to perform channel detection. Finally, we introduce a channel coefficient to select the channel with the best channel quality. Experimental results show that EDChannel algorithm achieves higher accuracy of channel prediction than ARIMA, linear regression(Linear) and autoregression(AR), and the overall throughput in stable environment and unstable environment is also increased by about 2.9% and 17.4% compared with random frequency hopping in Blink [17].

In the rest of this paper, Sect. 2 shows the related work. Sect. 3 gives the detailed design of the system. Sect. 4 describes the specific implementation details and data collection and evaluates the implemented methodology. And we conclude our work in Sect. 5.

2 Related Work

This section firstly introduces the related work of wireless channel prediction. Wireless channel prediction can be roughly classified into two categories: One is the Markov model, which models the channel changes by making simplified assumptions, relying on few network parameters and limited historical data; the other is a machine learning model, which is a channel prediction model used to predict the quality of wireless channels through historical data and machine learning algorithms [16]. We then introduce the related work of channel prediction based on deep learning in the backscatter communication network, and briefly describe the research status of channel detection and channel selection in the backscatter communication network. Finally, aiming at the shortcomings of the existing methods, we design our framework.

Channel prediction based on Markov model: Smith et al. proposed a linear finite state Markov predictor for channel prediction, and achieved a good balance between complexity and accuracy [18]. Halaseh et al. proposed a method based on a continuous hidden Markov model for time-frequency spectrum occupancy prediction in cognitive radio networks [19]. Traditional channel prediction methods based on Markov models have specific advantages when computing power is precious and data is limited. However, they usually assume that the transmitter knows all the channel state information and the receiver has error-free channel estimation, and these assumptions are usually difficult to meet in the actual channel prediction system. In addition, Wang et al. found that when using Markov models for channel prediction, only high-order Markov models can achieve better performance than low-order Markov models [20]. However, the high-order Markov model requires a higher amount of calculation, which further restricts the application of the Markov model in channel prediction.

Channel prediction based on machine learning: In the channel prediction of backscatter communication networks, previous work mainly focused on the use of machine learning for channel parameter estimation. For example, Wang's team used blind channel estimators based on EVD, EM algorithms, and LS algorithms to estimate the channel parameters when tags are in different states. And through the Cramer-Rao Lower Bounds lower bounds to evaluate the performance of the algorithm [9–11, 21, 22]. Darsena et al. deduced a computing feasible space alternate generalized expectation-maximization algorithm for joint channel estimation, interference cancellation and data detection of backscatter information [23]. While methods based on machine learning for channel prediction can give accurate channel parameters, they in some cases struggle to accurately identify dependencies between data points and require human intervention to maintain a better prediction. Instead, deep learning can automatically learn relationships from historical data by training on large amounts of data.

Channel prediction based on deep learning: In [6], the authors demonstrated that the use of a multi-layer BP neural network can accurately predict channel parameters, and evaluated and compared the throughput of the overall framework of BLINK [17] and CARA [24] in different environments. Although the channel prediction accuracy of this method is very high, the length of the channel indicator sequence predicted by it is relatively short.

Channel detection: The related work of channel detection in the backscatter communication network includes that BLINK [17] determines whether to perform detection with the trigger detecting the change of the position or movement pattern of the sensor tag. Then, only one probe is used to obtain channel information. Tian decides whether to perform channel detection or not by the trigger that can detect the movement of the sensor tag and the data of the acceleration sensor [6].BLINK and Tian's method can trigger channel detection in most cases promptly. However, in the experiments in section 3.3, we found that the received signal strength indicator continued to change significantly when only the obstacle moved and the tag did not make a move. We still need to perform channel detection on this situation to get real-time channel conditions. In addition, these two methods also require related hardware such as sensors to be implemented, which may not be sufficient in a real environment. CARA proves the channel correlation, and grouping channels through channel correlation. Then, multiple probes are used at the same time to reduce the overhead of channel detection [24]. Although the method of using multiple probes can quickly detect the state of the channel, it occupies limited channel resources.

Channel selection: In the backscatter communication network, the correct channel selection can improve the efficiency of information transmission and reduce the loss of information. In the existing channel selection research, Blink and CARA select directly through the channel metrics detected [17, 24]. Although this is convenient, it faces a certain degree of time delay. Tian selects channels based on predicted channel metrics [6]. Although the predicted value of the channel indicator can be obtained by BP neural network, the predicted sequence of the channel indicator is short and the amount of information about the channel is small.

Our approach: In this article, we design EDChannel for sequence-to-sequence channel prediction. In addition, we consider a hardware-free channel detection scheme for channel prediction in backscatter communication networks, and introduce a channel coefficient for channel selection. The accuracy and wide applicability of the EDChannel algorithm is demonstrated through the results of channel prediction of the EDChannel algorithm and the comparison algorithm in a variety of real-world situations. The comparison of

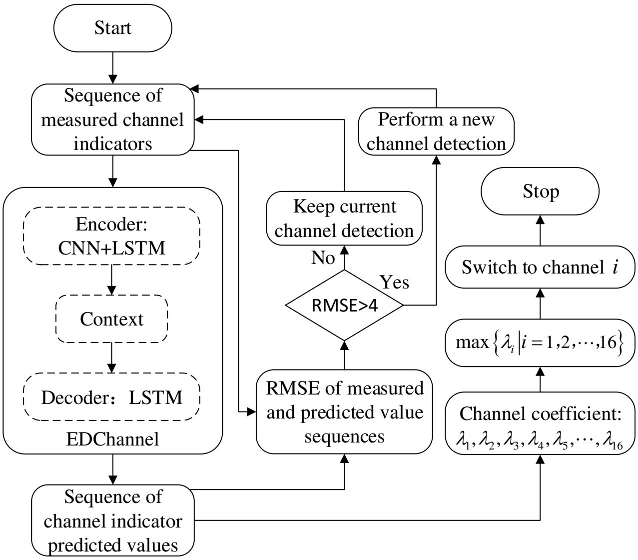


Fig. 1 System overview.

channel selection with random frequency hopping in stable and unstable environments demonstrates that channel selection provides an improvement in system throughput.

3 System Design

3.1 System Overview

In order to improve the performance of the backscatter communication system and the accuracy of channel selection, we propose a more accurate channel prediction framework and build a sequence-to-sequence deep learning model based on encoder-decoder, as shown in Fig. 1. Among them, the EDChannel encoder uses the CNN model for feature extraction of sub-sequences, and the LSTM model to help extract features across time steps. The decoder uses the LSTM model to be responsible for reading and interpreting the input sequence model to achieve sequence-to-sequence channel prediction. On the basis of channel prediction, we design the channel detection module and channel selection module. The channel detection module uses the RMSE of the measured value sequence of the channel indicator and the predicted value sequence as the trigger of the channel detection. The channel selection module uses the mean and variance of the normalized sequence of predicted values to determine the channel selection criteria and to measure the signal strength and stability of the channel respectively.

3.2 Channel prediction module

We design EDChannel, as shown in Fig. 2.

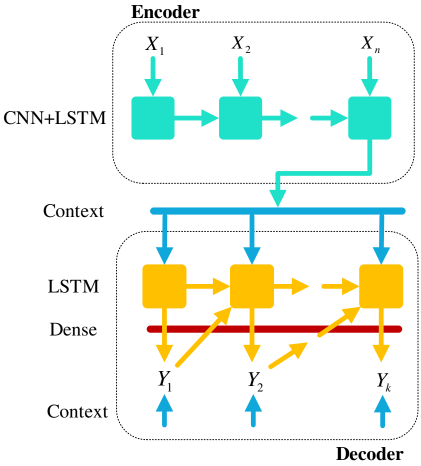


Fig. 2 Encoder-decoder structure diagram.

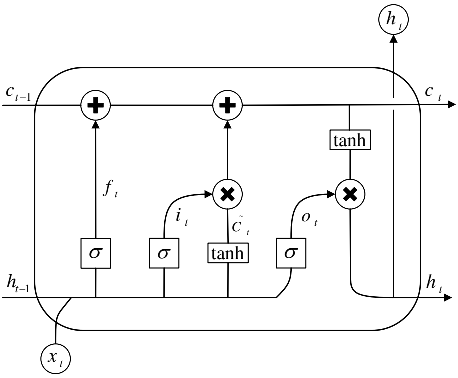


Fig. 3 LSTM unit structure diagram.

The encoder is responsible for reading and interpreting the input sequence model, it receives the past signal strength measurement value sequence $X = (X_1, X_2, \cdots, X_n)$, and generates a fixed-length context vector. It compresses and encodes the information of the entire input sequence X, and represents the model's interpretation of the channel strength measurement value sequence. The decoder is a model in charge of explaining each step in the generated output sequence. It uses the context vector output by the encoder and the output Y_{t-1} of the previous time step as input, and then outputs the signal strength prediction value sequence $Y_{t-1} = (Y_1, Y_2, \cdots, Y_k)$ [25]. Then connect a fully connected layer behind the decoder to share the weight.

We use LSTM units as the underlying neural network architecture in each layer of the encoder and decoder. As shown in Fig. 3, LSTM is a special recurrent neural network (RNN), which has the same structure of neural network repeating module chain like RNN, but the internal structure of repeating modules is different. Compared with the simple layer of RNN, LSTM has four layers, which interact especially [26, 27].

Fig. 4 CNN unit structure diagram.

LSTM solves the problem of vanishing or exploding gradients by adding gates for controlling access to past information and has been proved effective in many prediction tasks, such as wind speed prediction. Encoders and decoders retain important features through various gate functions in LSTM to ensure that important features will not be lost during long-term transmission. There are three types of gates: input gate i_t, forget gate f_t and output gate o_t. The calculation formula is as follows:

$$\begin{cases} f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \\ i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \\ \widetilde{C_t} = \tanh(W_c * [h_{t-1}, x_t] + b_c) \end{cases} \tag{1}$$

where W_f, W_i and W_c represent the corresponding weights. b_f, b_i and b_c represent the corresponding bias terms. $[h_{t-1}, x_t]$ represents the connection of two vectors into a longer vector, and σ is the sigmoid function. The forget gate determines how much of the cell state C_{t-1} from the previous moment is retained to the current moment C_t. The input gate determines how much of the network's input x_t at the current moment is saved in the unit state C_t. Then update the old unit state C_{t-1} to the new unit state C_t. Finally, calculate the output gate o_t and the unit output h_t of the LSTM. The calculation formula is as follows:

$$\begin{cases} C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \\ o_t = \sigma * (W_o * [h_{t-1}, x_t] + b_o) \\ h_t = o_t * \tanh(C_t) \end{cases} \tag{2}$$

Although LSTM has been proved in dealing with the time dependence of channel prediction, maintaining structural locality and solving such expansion problems remain to be challenged [28]. Therefore, in order to extract features completely, the CNN+LSTM discussed by Trigeorgis and Ringeval [29] is considered as an encoder in this paper. This model has been applied to traffic speed prediction and energy consumption prediction.

Fig. 4 shows the overall architecture of a CNN. Among them, each node of the convolution layer extracts features from the input sequence through convolution operation, and generates a feature map. The pooling layer compresses the input feature map, on the one hand, it makes the feature map smaller to simplify the network calculation complexity. On the other hand, feature compression is performed to extract the main features. The flattening layer mainly converts the three-dimensional layer in the network into a one-dimensional vector to fit the input of LSTM for long-term learning.

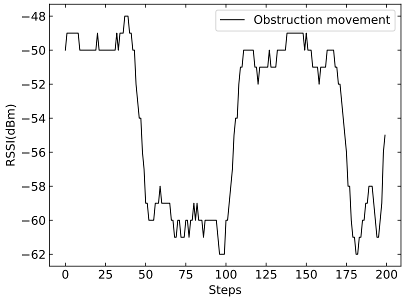


Fig. 5 The received signal strength changes only when the obstacle is moving.

3.3 Channel detection module

In the study of channel detection, we found that with only the obstacle moving and the tag not moving, the indicator of the channel received signal strength still changed significantly, as shown in Fig. 5. In this case, we still need to perform channel detection to get real-time channel conditions. At the same time, considering the design of our channel prediction module, we use the RMSE of the target tag predicted value sequence and the true value sequence as the trigger for channel detection. From a large number of preliminary experiments, we found that the RMSE value of the predicted value and the true value of the entire sequence is generally less than 5 when the signal fluctuation is predictable. So in order not to miss any better channels, we set the threshold at 4. This means that we do not perform channel detection when the RMSE value of the whole sequence prediction and the true value is less than 4. When the RMSE value is greater than 4, a new channel detection scheme is executed.

3.4 Channel selection module

Following the channel detection and prediction module, a sequence of channel indicator predictions is obtained, using the mean $\overline{X}$ and variance S^2 to measure the average received signal strength and stability of the predicted sequence, respectively. The specific calculation formula is as follows:

$$\begin{cases} \overline{X} = \dfrac{\sum_{i=1}^{n} X_i}{n} \\ S^2 = \dfrac{\sum_{i=1}^{n} (X_i - \overline{X})^2}{n} \end{cases} \tag{3}$$

where X_i is each value of the prediction sequence, and n is the number of data in the prediction sequence. In channel selection, we always prefer the mean value to be as large as possible so that the received channel strength will be greater. At the same time, we also hope that the variance is as small as possible, because the channel will be more stable in this way. However, in the course of our experiments, we found that when we chose the channel with the highest mean value, the variance of that channel was not always the smallest. This means that the channel chosen is not always the strongest and most stable at the same time. We therefore introduce a channel coefficient γ to indicate the channel's strength. The calculation formula of γ is as follows:

$$\begin{cases} \alpha = \dfrac{\overline{X}_j - \overline{X}_{min}}{\overline{X}_{max} - \overline{X}_{min}} \\ \beta = \dfrac{S^2_{max} - S^2_j}{S^2_{max} - S^2_{min}} \\ \gamma = A\alpha + B\beta \end{cases} \tag{4}$$

where $\overline{X}_j$ and S^2_j are the mean and variance of the corresponding sequence, $\overline{X}_{max}$ and $\overline{X}_{min}$ are the maximum and minimum mean values in all the sequences, S^2_{max} and S^2_{min} are the maximum and minimum variances in all the sequences. α is the positive normalization of the mean of the corresponding sequence, β is the inverse normalization of the variance of the corresponding sequence, and A and B are the impact factors.We think that α and β are equally important, so we set 0.5 for both. In the channel selection process, we always choose the channel with the maximum channel coefficient γ. When multiple channels have the same γ value, the channel with the largest α value is selected. And when two channels have exactly the same α and β, the channel with a frequency close to that of the channel currently being communicated is selected.

4 Implementation and evaluation

In this section, we use RFID to conduct a lot of experiments to evaluate the performance of our scheme. First, we introduced the experimental equipment, the data collection environment and three evaluation metrics. Then the optimal training data window and optimal step size of the EDChannel algorithm under unstable environment are discussed, and various initialization parameters of the experiment are given. Next, we carried out the prediction experiment of the EDChannel algorithm in a stable environment, and then discussed the experimental results of the car moving scene and the pedestrian moving scene. In order to make the evaluation more comprehensive, we used two additional metrics to analyze the experimental results, which proved the superior performance and wide applicability of the EDChannel algorithm. Finally, our experiments comparing the two channel selection methods show that channel selection using the channel coefficients of the prediction sequence can improve the throughput of the network.

4.1 Experimental equipment and data collection

Fig. 6 shows that Impinj Speedway R420 reader and two different tags as backscatter nodes in the network. The specific equipment parameters are shown in Table 1. Channel data was collected in a variety of environments as

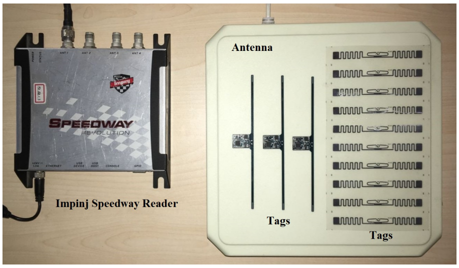
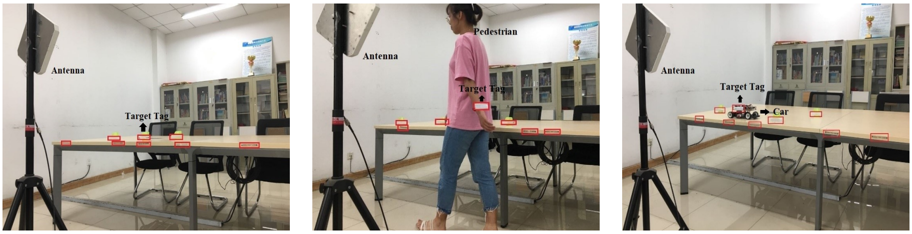


Fig. 6 Experimental equipment.

shown in Fig. 7 and a deep learning module based on Tensorflow 2.4.0 and Keras 2.4.3 was used to implement our model.



(a) (b) (c)

Fig. 7 Data collection environment display. (a) Stable environment. (b) Pedestrian moving scene. (c) Car moving scene.

Table 1 Device parameters

Parameters	Values
Versions	Keras versions = 2.4.3
	Tensorflow versions = 2.4.0
Reader	Model = Speedway R420
	Operating Region = China 920-925MHz
	Antennas = 1
	Tx power = 30dBm
	Rx sensitivity = -70dBm
Tags	Common RFID electronic tags
	WISP tags

We collect training data in both stable and unstable environments. Stable environment refers to the situation where the position of the node remains unchanged and there is no interference in the surrounding environment. An unstable environment refers to a situation where a node moves or is blocked or there is other signal interference around it. In the stable environment, we consider different distances and tag types. As shown in Fig. 7(a), we arranged 7 interference tags and 1 target tag, and collected the RSSI value of the target tag channel for 20 minutes. In the unstable environment, we considered the pedestrian movement scene and the car movement scene, as in Figs. 7(b), 7(c), respectively. We attached the target tags to the wrist of the pedestrian and the side of the car, and then collected the RSSI values of the channels for 20 minutes in an environment with 6 and 8 interfering tags, respectively. In the experiment, the pedestrian walks back and forth between the antenna and the table. The car moves along the table at a speed of approximately 0.25m/s back and forth. It is a challenge for our prediction task because of the huge variation in data in unstable environments. Therefore we collect more data in unstable environments than in stable ones.

4.2 Evaluation index

The main metrics used for evaluation are RMSE, Mean Absolute Error (MAE) and Relative Error (RE). RMSE and MAE capture the error in the absolute prediction, and RE captures the ratio of the error in the prediction to the actual channel change. The specific calculation formula is as follows:

$$\begin{cases} RMSE_j = \sqrt{\dfrac{\sum_{i=1}^{m}(\hat{y_{ij}} - y_{ij})^2}{m}} \\ MAE_j = \dfrac{\sum_{i=1}^{m}|\hat{y_{ij}} - y_{ij}|}{m} \\ RE_j = \dfrac{\sum_{i=1}^{m}\frac{|\hat{y_{ij}} - y_{ij}|}{y_{ij}}}{m} \end{cases} \tag{5}$$

where $\hat{y_{ij}}$ and y_{ij} are the predicted value and the true value of the i test sample of the j prediction step, and m is the number of test samples. In most papers related to channel prediction, the researchers mainly use RMSE as an evaluation method [16]. For consistency, we also use RMSE as the main evaluation metric for this paper, supplemented by MAE and RE.

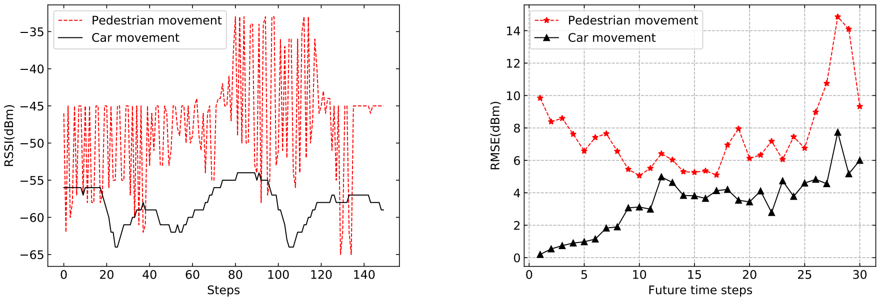
4.3 The effect of prediction step size and training data length

In the following, we discuss the effect of the prediction step size and training data length of the EDChannel algorithm on the prediction accuracy. Since the channel prediction effect of EDChannel in a stable environment is much better than that in an unstable environment, we will mainly discuss the prediction step size and training data length in an unstable environment below.

A. The effect of prediction step size

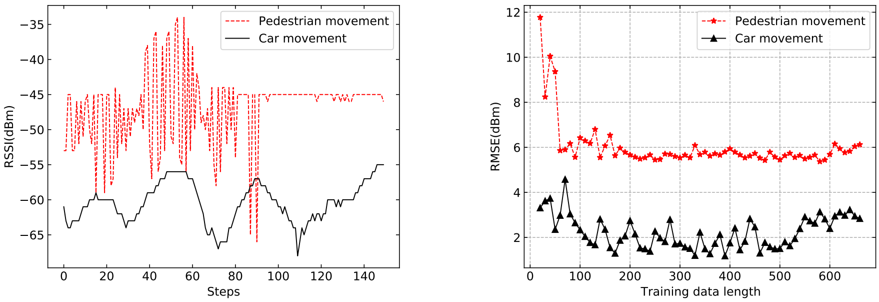
We use a controlling variable approach to discuss the effect of the prediction step size of the EDChannel algorithm on the prediction accuracy. We first fixed the length of the training data to 600 and the number of iterations to 500-5000, then we varied only the prediction step size. The range of prediction step size was 1-30, each step size was added 1 compared to the previous one, and the experiment was repeated 3 times for each same step size. At the end, the differences in RMSE values at different step sizes were evaluated together.

Fig. 8(a) shows the variation of the received signal strength in a real environment. We find that the data fluctuates more steadily in the car movement scene, while the data fluctuates more dramatically in the pedestrian movement



(a) (b)

Fig. 8 Effect of prediction step size on prediction accuracy. (a) Variation of received signal strength in real scenes. (b) RMSE values corresponding to different prediction steps.



(a) (b)

Fig. 9 Effect of training data length on prediction accuracy. (a) Variation of received signal strength in real scenes. (b) RMSE values corresponding to different training data lengths.

scene. Fig. 8(b) shows the variation of RMSE values from 1 to 30 prediction steps for each of the two scenes. We found that the RMSE value of EDChannel increases slowly with the increase of the time step in the scene of moving the car. In a pedestrian moving scene, the RMSE value of EDChannel is very high when the step length is short. But as the step size increases, the RMSE value begins to decrease. And until the step size increases to 10, the RMSE value starts to increase with the increase of the step size. The main reason is that the data fluctuates greatly when the data is unstable. However, as a whole, EDChannel's RMSE values in both cases continue to increase slowly with the predicted time step.

B. The effect of training data length

We still use the controlling variable approach, fix the prediction step size to 10, and the number of iterations to 500-5000. Then, we varied the training data length only. The range of training data is 10-660, each training data length is added 10 compared to the previous one, and the experiment is repeated 3 times for each same training data length.

Fig. 9(a) shows the variation of the received signal strength in a real environment. We find that the data fluctuates more steadily in the car movement scene, while the data fluctuates more dramatically in the pedestrian movement scene. Fig. 9(b) shows the variation of RMSE values from 10 to 660 for the training data length in the two scenes, respectively. In the car movement scene, the RMSE value of EDChannel is poorly predicted when the training data is small. The reason for this is mainly that the training data cannot meet the generalization requirements of the EDChannel model, and the variety of fluctuation cases of the training data is smaller than the variety of fluctuation cases of the predicted data, which cannot fully learn the data fluctuation cases. However, as the data volume of training data increases, the RMSE of EDChannel is limited to fluctuate around 2, and the prediction effect can meet the expected requirements. In the pedestrian movement scene, we found that the RMSE value of EDChannel is very poor in prediction when the training data is small. However, as the training data increases, the RMSE of EDChannel is limited to fluctuate between 5 and 6, and the prediction results can meet the expected requirements.

C. Summary

Fig. 8(a) and Fig. 9(a) shows the received signal strength variation in a real scene. We found that the data fluctuations in the scene of the car movement are relatively stable, while the data fluctuations in the scene of the pedestrian movement are more severe. From Fig. 8(a) and Fig. 8(b), we found that setting the prediction step size at 10 is more suitable for both the car movement scene and the pedestrian movement scene, and the computational overhead is also small. However, it is worth noting that the longest stay time of a single channel of the backscatter communication network varies according to regional regulations. For example, in North America, the FCC allows a single channel to reach 0.4 seconds in 10 seconds. In China, the maximum stay time is 2 seconds [6]. Therefore, we set the prediction time to 2 seconds, which means that the prediction is re-predicted every 2 seconds. At the same time, we can predict the data changes in the next 2 seconds through multi-step forecasting. From Fig. 9(b), we found that the training data length is between 200 and 600, and the fluctuation of the RMSE value is relatively stable, which is more suitable for the scenes of car movement and pedestrian movement. But from Fig. 8(a) and Fig. 9(a), we found that in an unstable environment, the channel quality changes quickly. So in order to collect more training data, we set the training data length between 300 and 500. In this way, historical data can be fully learned to meet the prediction of future data, and the computational overhead can be controlled within a reasonable range to meet the high-speed iteration of the model. In the actual data collection process, our equipment can collect approximately 39 data points per second. Therefore, to predict the data for the next 2 seconds when the step length is 10, we need to predict about 8 steps. The training data length of 300 to 500 requires us to continuously collect about 7.5s to 13s for a channel.

4.4 Initialization parameters

When using new data to execute the EDChannel algorithm, we need to repeatedly estimate the parameters, which is very low in computational efficiency. For the sequence-to-sequence prediction model of channel metrics for backscattered communication networks, we give the initialisation parameters for the EDChannel model, as shown in Table 2. Hence we can improve the computational efficiency by initialising the parameter search.

As mentioned earlier (Section 3.2), in EDChannel, both the encoder and decoder use deep RNNs. The encoder uses CNN+LSTM as the basic unit, and the decoder only uses LSTM as the basic unit. We observe that in order to provide the best performance, the optimal parameter configuration of EDChannel varies with the consideration of the data set. An encoder or decoder is composed of 1 or 2 layers of basic unit stacks, and each layer has 20 to 100 hidden units. The 1D CNN layer uses 8, 16 or 32 filters, 1 convolution kernel, 1 max-pooling layer, and the activation function is ReLU. The number of neurons in the LSTM layer stacked with the CNN and the LSTM layer as the decoder are both 20-100, and the activation function is also ReLU. The objective function of the model is MSE, and the optimizer is Adam. It usually requires 500-10000 times of training. By balancing performance and training time, the number of times is selected based on experience.

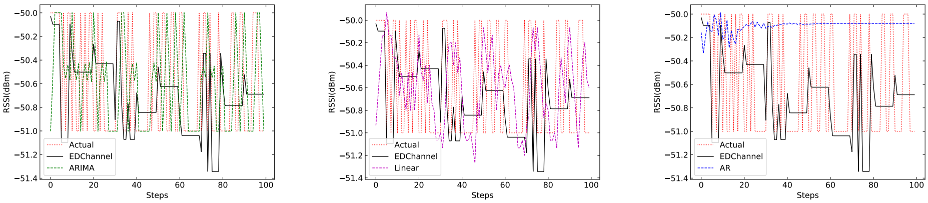
Table 2 Device parameters

Parameters	Values
Conv1D	Filters = 8/16/32
	Convolution kernel size = 1
	Activation = Relu
	Max pooling size = 1
LSTM	The number of neurons = 20-100
	Activation = Relu
Training	Optimizer = Adam
	Loss function = MSE
	Max.number of epoches = 500-10000
	Data length = 300-500
Prediction	Step length = 10
	Data length = 100

4.5 Channel prediction evaluation in stable environment

We will firstly compare the performance of EDChannel with Linear, AR, and ARIMA in a stable environment. The AR and ARIMA (p, d, q) algorithms also consider the history of 300-500 samples in the past to predict the future 10 data and predict the data values of 10 time steps. And Linear considers the history of the past 10 samples to fit the data values of the future 10 samples, and also predicts ten steps.

The RSSI results of the real value and the predicted value of EDChannel, Linear, AR and ARIMA are shown in Fig. 10. The RSSI value in a stable environment tends to be stable for a long time, fluctuating between -51 and



(a) (b) (c)

Fig. 10 Comparison of the true value and the predicted value in a stable environment. (a) EDChannel and ARIMA. (b) EDChannel and Linear. (c) EDChannel and AR.

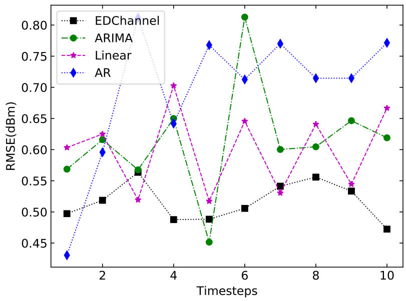


Fig. 11 RMSE of predicted channel data for stable environment.

-50. And the predicted value and true value of EDChannel and the three comparison algorithms are not much different. From Fig. 11, we found that the RMSE value between the predicted value and the true value of EDChannel is always less than 0.6. Compared with Section 4.6, the prediction effect under the unstable environment is excellent. The experimental results show that our method exhibits excellent performance in a stable environment.

4.6 Channel prediction evaluation in unstable environment

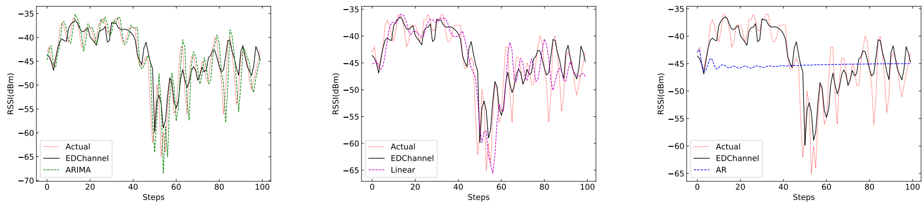
In this section, we will compare the performance of EDChannel with Linear, AR, and ARIMA in unstable environments. The parameter settings of the three algorithms are the same as in Section 4.5.

A. Channel prediction for car moving scene

Fig. 12 shows the comparison between the predicted value of RSSI and the true value in the scene of a moving car. We found that the prediction results of EDChannel, ARIMA and Linear are more consistent with the real situation. The AR gradually loses its predictive function at 20 steps and is less effective in prediction. From Fig. 13, we found that the RMSE of the EDChannel algorithm at most time steps is less than that of the three comparison algorithms. And as the time step increases, the accuracy of the EDChannel algorithm is still high.

B. Pedestrian movement scene channel prediction

Fig. 14 shows the comparison between the predicted value of RSSI and the true value in a pedestrian moving scene. We found that the prediction effect of



(a) (b) (c)

Fig. 12 Comparison of the true value and the predicted value in the moving scene of the car. (a) EDChannel and ARIMA. (b) EDChannel and Linear. (c) EDChannel and AR.

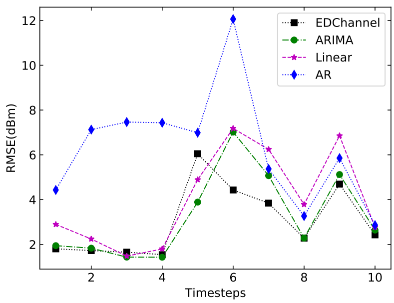
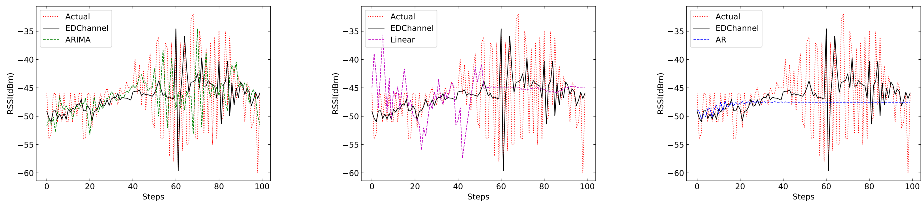


Fig. 13 The RMSE of the predicted value and the true value of the channel data in the moving scene of the car.



(a) (b) (c)

Fig. 14 Comparison of real and predicted values in pedestrian movement scenes. (a) EDChannel and ARIMA. (b) EDChannel and Linear. (c) EDChannel and AR

ARIMA and EDChannel is better, while the prediction effect of Linear in the initial stage is poor. The AR algorithm gradually loses its predictive function at 30 steps, and the predictive effect is poor. From Fig. 15, we found that the RMSE of the EDChannel algorithm is smaller than the comparison algorithm at most time steps. And as the time step increases, the EDChannel algorithm can still maintain the accuracy of the prediction.

C. Summary

In summary, from Fig. 12(c) and Fig. 14(c), we found that the AR algorithm is effective when the prediction step is short. However, as time goes by, the predictive performance of AR algorithms gradually deteriorates. This shows that the information extracted by the AR algorithm in the past 300-500

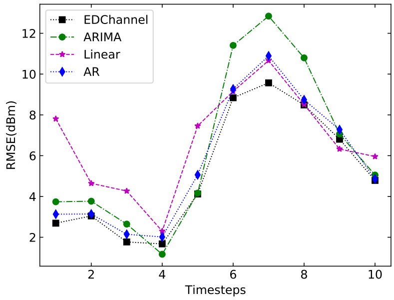


Fig. 15 The RMSE of the predicted value and the true value of the channel data in the pedestrian moving scene.

samples is not enough to support multi-step prediction. Since AR failed to capture the time correlation of the wireless channel, the prediction performance is poor. In the following experiments, we will only compare the prediction results of EDChannel, Linear and ARIMA. In Fig. 13 and Fig. 15, we observed that the performance of the EDChannel algorithm is more competitive than Linear and ARIMA in general. In addition, compared with Linear and ARIMA, the RMSE value of EDChannel increases more slowly with the time step. It means that EDChannel can predict the future better than other algorithms. But the predictive performance of all models decreases as they go further into the future predictions.

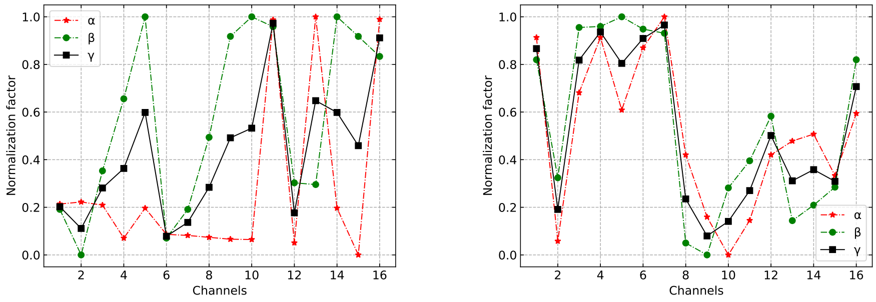
4.7 Results of RE and MAE

The following we use two other evaluation metrics, RE and MAE, for additional illustration of the experiment. Table 3 shows the results of RE and MAE of EDChannel algorithm, ARIMA and Linear algorithm in a variety of environment prediction data and real data.

Table 3 Results of RE and MAE of EDChannel, ARIMA, and Linear algorithms in different environments

Environment	RE			MAE		
	EDChannel	ARIME	Linear	EDChannel	ARIME	Linear
Stable environment	0.836	0.992	1.015	0.423	0.502	0.513
Car moving scene	5.110	6.178	7.438	2.279	2.756	3.317
Pedestrian moving scene	9.748	12.069	12.549	4.524	5.601	5.824

We found that the performance of the EDChannel algorithm is also better than the comparison algorithm under the evaluation of the RE and MAE metrics. Moreover, the performance improvement of EDChannel in unstable environments is more obvious than that of ARIMA and Linear. In general, experiments have proved the accuracy and applicability of EDChannel in predicting received signal strength in a variety of environments.



(a) (b)

Fig. 16 Normalized mean, variance and channel coefficient of 16 channels. (a) Stable environment. (b) Unstable environment.

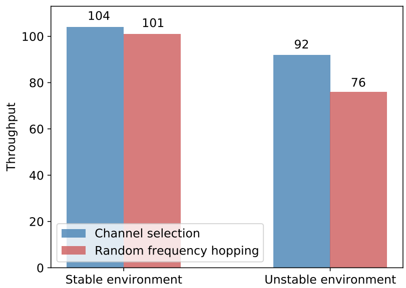


Fig. 17 Comparison of throughput between channel selection and random frequency hopping in stable and unstable environments

4.8 Channel selection through channel prediction

In the following we evaluate the effectiveness of our prediction algorithm through simulation experiments and five passive tags. In order to obtain experimental data, we use the reader to read the data of each channel in turn under the same circumstances, and collect the RSSI value of each channel. In a stable environment, it is easier to keep the environment unchanged and collect 16 channels of data. Under unstable conditions, however, it is more difficult to ensure that the tag is in the same state of motion for each experiment. To ensure that the conditions are consistent from one collection to the next, we attach the labels to the pendulum and release them from the same height each time. We consider that the errors under these experimental conditions do not affect the experimental results. Then, we model and predict each channel separately, and get the prediction sequence of each channel for a period of time. Finally, we select the channel according to the channel coefficient γ of each channel prediction sequence. And compare our method with random frequency hopping.

In Fig. 16, channel 11 and channel 7 have the largest channel coefficients in stable and unstable environments, so we choose channel 11 and channel 7 respectively as the communication channel for the next time period. Then, we compare the channel selected according to γ with the channel selected by random frequency hopping in each of the two cases. The comparison result of the throughput of communication which is 4 s is shown in Fig. 17. In contrast to random frequency hopping, channel selection in a stable environment has a small increase in throughput, which is only increased by about 2.9%. We attribute it to the fact that each channel in a stable environment is better, and the channel selection does not show enough superiority. Compared with random frequency hopping, channel selection in an unstable environment improves throughput more significantly, increasing by about 17.4%. On the one hand, because of the channel changing drastically in an unstable environment, channel selection can maintain a better channel for communication. On the other hand, it is because random frequency hopping can avoid difficult channels though. However, it is less stable due to randomness. The experimental results show that the method of channel selection through channel prediction can effectively increase the network throughput.

5 Conclusion

In this paper, we design a deep learning channel prediction method for the problem of channel prediction in backscatter communication networks. Firstly, we developed EDChannel, a sequence-to-sequence deep learning model based on encoder-decoder. It predicts future changes in channel metrics by learning from changes in previous channel metrics. Secondly, the RMSE of the sequence of measured values and the sequence of predicted values is used as a trigger for channel detection to determine whether to perform a new channel detection. Finally, channel selection is performed based on channel coefficients composed of mean and variance. Experiments show that EDChannel outperforms the comparative algorithms ARIMA, Linear, and AR in terms of interference resistance and applicability in a wide range of environments. Our prediction method improves the throughput of the network by approximately 2.9% and 17.4% in stable and unstable environments compared with the widely used random frequency hopping.

Declarations

- Funding This paper supported by The General Object of National Natural Science Foundation under Grants(61972273); National Major Scientific Research Instrument Development Project (6202780085); National Natural Science Foundation of China under Grant(62102280); Applied Basic Research Program of Shanxi Province(20210302124167); National Key R&D Project(2018YFB2200900); Shanxi Province key technology and generic technology R&D project(2020XXX007).
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use) No conflict of interest
- Consent for publication Yes

References

[1] J. Mitsugi, Y. Kawakita, K. Egawa, and H. Ichikawa, "Perfectly synchronized streaming from multiple digitally modulated backscatter sensor tags," *IEEE Journal of Radio Frequency Identification*, vol. 3, no. 3, pp. 149–156, 2019.

[2] L. Ma, K. Hu, Y. Zhu, and H. Chen, "Cooperative artificial bee colony algorithm for multi-objective rfid network planning," *Journal of Network and Computer Applications*, vol. 42, pp. 143–162, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804514000551

[3] Q. Wang, J. Zhao, S. Xu, and R. Wang, "Retype: Your breath tells your mind!" *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 9436–9449, 2021.

[4] Q. Miao, F. Xiao, H. Huang, L. Sun, and R. Wang, "Smart attendance system based on frequency distribution algorithm with passive rfid tags," *Tsinghua Science and Technology*, vol. 25, no. 2, pp. 217–226, 2020.

[5] J. Zhao, J. Li, D. Li, and H. Yang, "Optimal data transmission in backscatter communication for passive sensing systems," *Tsinghua Science and Technology*, vol. 25, no. 5, pp. 647–658, 2020.

[6] J. Zhao, H. Tian, and D.-a. Li, "Channel prediction based on bp neural network for backscatter communication networks," *Sensors*, vol. 20, no. 1, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/1/300

[7] W. Gong, S. Chen, J. Liu, and Z. Wang, "Mobirate: Mobility-aware rate adaptation using phy information for backscatter networks," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1259–1267.

[8] L. Deng and J. W. Mark, "Parameter estimation for markov modulated poisson processes via the em algorithm with time discretization," *Telecommunication Systems*, vol. 1, no. 1, pp. 321–338, 1993.

[9] W. Zhao, G. Wang, S. Atapattu, and B. Ai, "Blind channel estimation in ambient backscatter communication systems with multiple-antenna reader," in *2018 IEEE/CIC International Conference on Communications in China (ICCC)*, 2018, pp. 320–324.

[10] S. Ma, Y. Zhu, G. Wang, and R. He, "Machine learning aided channel estimation for ambient backscatter communication systems," in *2018 IEEE International Conference on Communication Systems (ICCS)*, 2018, pp. 67–71.

[11] S. Ma, G. Wang, R. Fan, and C. Tellambura, "Blind channel estimation for ambient backscatter communication systems," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1296–1299, 2018.

[12] Y. Lv, S. Peng, Y. Yuan, C. Wang, P. Yin, J. Liu, and C. Wang, "A classifier using online bagging ensemble method for big data stream learning," *Tsinghua Science and Technology*, vol. 24, no. 4, pp. 379–388, 2019.

[13] H. Senol, A. R. B. Tahir, and A. Özmen, "Artificial neural network based estimation of sparse multipath channels in ofdm systems," *Telecommunication Systems*, vol. 77, no. 1, pp. 231–240, 2021.

[14] W.-S. Son and D. S. Han, "Analysis on the channel prediction accuracy of deep learning-based approach," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2021, pp. 140–143.

[15] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. [Online]. Available: https://aclanthology.org/D14-1179

[16] A. Kulkarni, A. Seetharam, A. Ramesh, and J. D. Herath, "Deepchannel: Wireless channel quality prediction using deep learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 1, pp. 443–456, 2020.

[17] P. Zhang, J. Gummeson, and D. Ganesan, "Blink: A high throughput link layer for backscatter communication," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 99C112. [Online]. Available: https://doi.org/10.1145/2307636.2307646

[18] J. Zhang, D. Smith, and Z. Chen, "Linear finite state markov chain predictor for channel prediction," in *2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications - (PIMRC)*, 2012, pp. 2085–2089.

[19] R. A. Halaseh and D. Dahlhaus, "Continuous hidden markov model based interference-aware cognitive radio spectrum occupancy prediction," in *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2016, pp. 1–6.

[20] H. S. Wang and P.-C. Chang, "On verifying the first-order markovian assumption for a rayleigh fading channel model," *IEEE Transactions on Vehicular Technology*, vol. 45, no. 2, pp. 353–357, 1996.

[21] S. Ma, G. Wang, R. Fan, and C. Tellambura, "Blind channel estimation for ambient backscatter communication systems," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1296–1299, 2018.

[22] C. Chen, G. Wang, F. Gao, Y. C. Eldar, and H. Guan, "Blind detection for ambient backscatter communication system with multiple-antenna tags," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 206–212.

[23] D. Darsena, G. Gelli, and F. Verde, "Joint channel estimation, interference cancellation, and data detection for ambient backscatter communications," in *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2018, pp. 1–5.

[24] W. Gong, H. Liu, K. Liu, Q. Ma, and Y. Liu, "Exploiting channel diversity for rate adaptation in backscatter communication networks," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9.

[25] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 1672–1678.

[26] C. T. Abdallah, D. Hush, and B. Horne, "Recursive neural networks for signal processing and control," in *Proceedings of the 1991 IEEE Workshop Neural Networks for Signal Processing [1991]*. IEEE, 1991, p. 523.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] S. Li, Q. Wang, X. Liu, and J. Chen, "Low cost lstm implementation based on stochastic computing for channel state information prediction," in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2018, pp. 231–234.

[29] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5200–5204.