Predictive Relay Selection: A Cooperative Diversity Scheme Using Deep Learning

Wei Jiang
German Research Center for Artificial Intelligence (DFKI)
Kaiserslautern, Germany
https://orcid.org/0000-0002-3719-3710

Hans Dieter Schotten
University of Kaiserslautern
Kaiserslautern, Germany
https://orcid.org/0000-0001-5005-3635

***Abstract*—In this paper, we propose a novel cooperative multi-relay transmission scheme for mobile terminals to exploit spatial diversity. By improving the timeliness of measured channel state information (CSI) through deep learning (DL)-based channel prediction, the proposed scheme remarkably lowers the probability of wrong relay selection arising from outdated CSI in fast time-varying channels. It inherits the simplicity of opportunistic relaying by selecting a single relay, avoiding the complexity of multi-relay coordination and synchronization. Numerical results reveal that it can achieve full diversity gain in slow-fading channels and substantially outperforms the existing schemes in fast-fading wireless environments. Moreover, the computational complexity brought by the DL predictor is negligible compared to off-the-shelf computing hardware.**

***Index Terms*—Cooperative diversity, outdated CSI, channel prediction, deep learning, LSTM, opportunistic relaying**

I. Introduction

Cooperative diversity [1] is an effective technique for mobile terminals without an antenna array to cultivate spatial diversity that is typically achieved by co-located multi-antenna systems. A main challenge of cooperative diversity is the inherent asynchronization among spatially-distributed antennas (relays). Multiple timing offset and multiple carrier frequency offset [2] among simultaneously-transmitting relays make *multi-relay* transmission such as distributed beam-forming and distributed space-time coding [3] too complicated for practical systems. In contrast, a *single-relay* approach called opportunistic relay selection (ORS) or opportunistic relaying [4] achieves full diversity gain while the complexity of multi-relay synchronization and coordination is avoided.

However, ORS is applicable only in slow-fading wireless environments since channel state information (CSI) used to select the best relay may be outdated quickly in fast-fading channels. Using a wrongly-selected relay substantially deteriorates the performance of ORS, as widely verified in the literature such as [5]–[7]. With the proliferation of high-mobility applications (such as vehicle-to-X, high-speed train, and unmanned aerial vehicle) and the utilization of higher frequency bands (e.g., millimeter wave and Terahertz) in 5G and beyond systems, the problem of outdated/aged CSI becomes more challenging. A cooperative method called generalized selection combining [8] shows robustness under aged channel but it suffers from a substantial loss of spectral efficiency. The authors of [9] proposed a method utilizing the knowledge of channel statistics, getting only a marginal gain, whereas the complexity obviously grows. By far, opportunistic space-time coding (OSTC) proposed by the author of this paper in [10]–[12] is the best method in fast fading channel from the perspective of diversity-multiplexing trade-off. But its performance gap to perfect selection using the perfect knowledge of channel is still large, motivating our follow-up works presented here.

In this paper, therefore, we propose a novel cooperative method coined predictive relay selection (PRS) for mobile terminals to exploit the gain of spatial diversity. The probability of wrong relay selection due to outdated CSI is remarkably reduced by improving the timeliness of CSI through fading channel prediction [13]–[21]. A deep recurrent neural network is deliberately built to provide high-accurate CSI predictions. The proposed scheme inherits the simplicity of ORS by selecting a single opportunistic relay to avoid the complexity of multi-relay coordination and synchronization. Simulation results reveal that it can achieve full diversity order in slow-fading channels and substantially outperforms the existing schemes in fast-fading wireless environments. Moreover, the computational complexity brought by the deep learning (DL)-based predictor is analyzed and compared with commercial off-the-shelf (COTS) computing hardware. The rest of this paper is organized as follows: Section II introduces the system model. Section III and IV present the proposed selection scheme and the principle of channel predictor, respectively. Complexity analysis and numerical results are given in Section V and VI. Finally, Section VII concludes this paper.

II. System Model

Following the working assumption applied for most of prior research works [2]–[12], we consider a dual-hop decode-and-forward (DF) cooperative network where a single source node s communicates with a single destination node d with the aid of K relays. Each node is equipped with a single antenna that is used for both signal transmission and reception over a narrow-band channel. The received signal in an arbitrary link $A \rightarrow B$ is modeled as $y_B = h_{A,B}x_A + z_B$, where $x_A \in \mathcal{C}$ is the transmitted symbol from node A with average power $P_A = \mathbb{E}[|x_A|^2]$, z_B stands for additive white Gaussian noise

*This work was supported by German Federal Ministry of Education and Research (BMBF) through TACNET4.0 project (Grant no. *KIS15GTI007*) and KICK project (Grant no. *16KIS1105*).

with zero-mean and variance σ_n^2, i.e., $z \sim \mathcal{CN}(0, \sigma_n^2)$, and $h_{A,B}$ represents the fading coefficient of the channel from A to B, which is a zero-mean circularly-symmetric complex Gaussian random variable $h \sim \mathcal{CN}(0, \sigma_h^2)$ under the assumption of Rayleigh fading. The instantaneous signal-to-noise ratio (SNR) is denoted by $\gamma_{A,B} = |h_{A,B}|^2 P_A / \sigma_n^2$ and the average SNR $\bar{\gamma}_{A,B} = \mathbb{E}[\gamma_{A,B}] = P_A \sigma_h^2 / \sigma_n^2$.

In a practical system, there exists a delay between the time of relay selection and the instant of using the selected relay to transmit signals. The actual CSI h may differ from its outdated version $\hat{h}$ that is applied for selecting relays. To quantify the quality of CSI, the correlation coefficient between h and $\hat{h}$ is introduced, i.e., $\rho_o = \frac{\mathbb{E}[h\hat{h}^*]}{\sqrt{\mathbb{E}[|h|^2]\mathbb{E}[|\hat{h}|^2]}}$. With the classical Doppler spectrum of the Jakes model, it takes the value

$$\rho_o = J_0(2\pi f_d \tau), \tag{1}$$

where f_d is the maximal Doppler frequency, τ stands for the delay between the outdated and actual CSI, and $J_0(\cdot)$ denotes the $zeroth$ order Bessel function of the first kind.

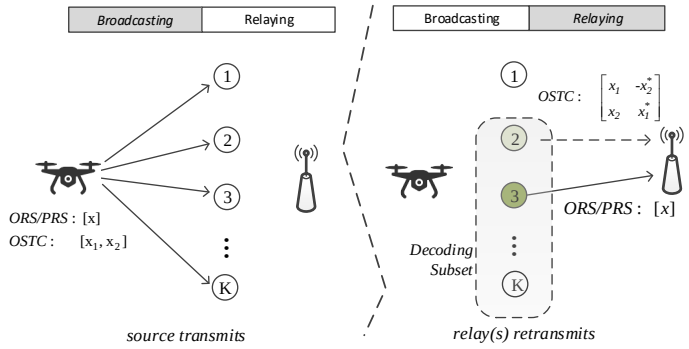


Fig. 1. Schematic diagram of a cooperative network using different DF relaying strategies: ORS, PRS, and OSTC.

Due to severe signal attenuation, a single-antenna relay should operate in half-duplex transmission mode to prevent from harmful self-interference between the transmitter and receiver. Therefore, its signal transmission is organized in two phases: the source broadcasts a signal in the source-to-relay (denoted by $\mathbb{SR}$ hereinafter) link, and then the relays retransmit this signal in the relay-to-destination ($\mathbb{RD}$) link. In the first phase, as shown in Fig.1, the source (e.g., the drone in the figure) sends a symbol x and those relays who can correctly decode x form a *decoding subset* ($\mathcal{DS}$) of the $\mathbb{SR}$ link

$$\mathcal{DS} \triangleq \{k \,|\log_2(1+\gamma_{s,k}) \geqslant 2R\}, \tag{2}$$

where R is an end-to-end ($\mathbb{EE}$) target rate for the dual-hop relaying. Note that the required data rate for either hop is doubled to $2R$ due to the adoption of half-duplex transmission. The best relay (denoted by $\dot{k}$) in ORS is opportunistically selected from $\mathcal{DS}$ in terms of $\dot{k} = \arg\max_{k \in \mathcal{DS}} \hat{\gamma}_{k,d}$, where $\hat{\gamma}_{k,d}$ is the SNR of the $\mathbb{RD}$ link *at the instant of relay selection*, which is an outdated version of the actual SNR $\gamma_{k,d}$ during signal transmission. In contrast, the proposed PRS scheme replaces the outdated CSI with the predicted CSI $\check{h}$, and determines $\dot{k}$ in terms of $\dot{k} = \arg\max_{k \in \mathcal{DS}} \check{\gamma}_{k,d}$, where $\check{\gamma}_{k,d} = |\check{h}_{k,d}|^2 P_k / \sigma_n^2$. In addition to the best relay, the OSTC scheme [10] needs another relay with the second strongest SNR, i.e., $\ddot{k} = \arg\max_{k \in \mathcal{DS} - \{\dot{k}\}} \hat{\gamma}_{k,d}$. In the first phase, the source broadcasts a pair of symbols (x_1, x_2) to all relays over two consecutive symbol periods. The regenerated signals are encoded by means of the Alamouti scheme [11], which is the unique space-time code achieving both full rate and full diversity, at the pair of selected relays. In the second phase, a relay transmits $(x_1, -x_2^*)$ while another transmits (x_2, x_1^*) simultaneously at the same frequency.

III. Predictive Relay Selection

Taking advantage of new degree of freedom opened by channel prediction, we propose the PRS scheme that can also achieve high performance in fast time-varying channels. The prediction horizon relaxes the tight requirement of time procedure and therefore provides the flexibility to design an advanced relaying strategy. As depicted in Algorithm 2, the implementation for PRS is detailed as follows:

1) At frame t, as illustrated in Fig.2, the source broadcasts a packet containing a pilot called Ready-To-Send (RTS) [4] and data payload. The CSI $h_{s,k}[t]$ is acquired at relay k by estimating RTS and is used for detecting data symbols. Those relays that correctly decode the source's signal comprise a $\mathcal{DS}$.
2) Clear-To-Send (CTS) is sent from the destination, so that relay k can estimate $h_{d,k}[t]$ and then $h_{k,d}[t]$ is known due to channel reciprocity. It feeds $h_{k,d}[t]$ into its embedded channel predictor to generate $\check{h}_{k,d}[t+1]$, and buffers it for its usage at the upcoming frame $t+1$.
3) Meanwhile, relay k belonging to the $\mathcal{DS}$ fetches $\check{h}_{k,d}[t]$ that is buffered at the previous frame $t-1$. This operation starts once the CTS arrives, in parallel with Step 2.
4) Then, a timer with a duration T_t proportional (denoted by $\propto$) to $1/|\check{h}_{k,d}[t]|$ is started at relay k.
5) The timer on the relay with the largest channel gain expires first, and then it sends a short packet to announce.
6) Once received the best relay's notification, other relays terminate their timers and keep silent. The selected relay forwards the signal until the end of this frame.

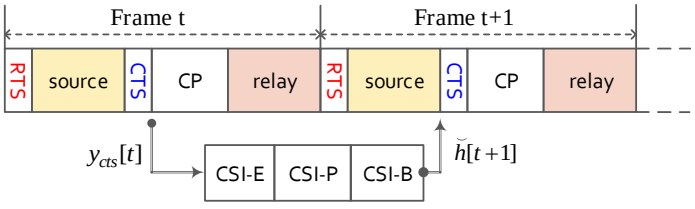


Fig. 2. Frame structure of PRS. *CSI-E: CSI Estimation, CSI-P: CSI Prediction, CSI-B: CSI Buffering, CP: Contention Period.*

Algorithm 1 Predictive Relay Selection

for $t = 1, 2, ...$ **do**
  s sends RTS; s sends data payload $\boldsymbol{x}[t]$
  for $k = 1, ..., K$ **do**
    estimate $h_{s,k}[t]$; $\hat{\boldsymbol{x}}[t] = f(\boldsymbol{y}_{s,k}[t], h_{s,k}[t])$
    if $\hat{\boldsymbol{x}}[t]$ is error-free **then**
      fetch $\check{h}_{k,d}[t]$; start a timer $T_t \propto \frac{1}{|\check{h}_{k,d}[t]|}$
    end if
  end for
  d sends CTS
  $\dot{k} = \arg\max_{k \in \mathcal{DS}} \left(|\check{h}_{k,d}[t]|\right)$ notifies its presence
  $\dot{k}$ transmits $\hat{\boldsymbol{x}}[t]$
  for $k = 1, ..., K$ **do**
    estimate $h_{k,d}[t]$; predict $\check{h}_{k,d}[t+1]$
    write $\check{h}_{k,d}[t+1]$ into Buffer
  end for
end for

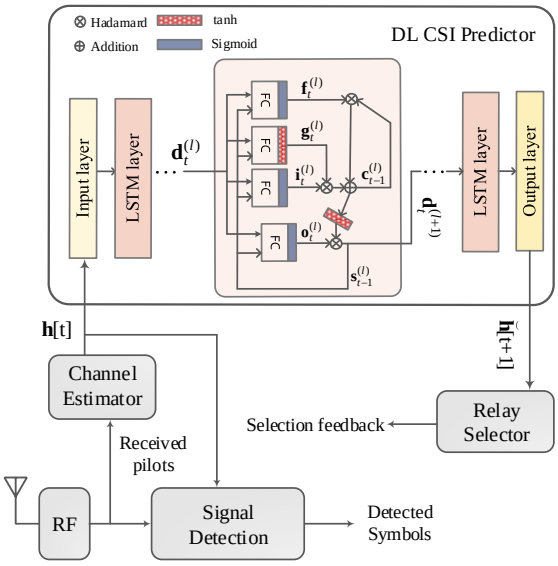


Fig. 3. Block diagram of the receiver of a relay in PRS. The DL-based predictor consists of an input layer, L LSTM hidden layers, and an output layer, where the l^{th} hidden layer is opened to illustrate the internal structure of an LSTM memory block.

IV. DL-based Channel Prediction

This section first introduces the principle of deep recurrent networks including simple recurrent neural network (RNN) [16], Long Short-Term Memory (LSTM) [22], and Gated Recurrent Unit (GRU) [23], followed by the explanation of applying a recurrent network to build a channel predictor.

A. Deep Recurrent Networks

Unlike uni-direction information flow in feed-forward neural networks, RNN has recurrent self-connections, which are applied to memorize historical states, exhibiting great potential in time-series prediction. The activation of the previous time step is fed back as part of the input for the current step. In a simple RNN, its l^{th} recurrent layer is generally modeled as

$$\mathbf{d}_t^{(l+1)} = \mathcal{R}^{(l)}(\mathbf{d}_t^{(l)}) = \delta_h\left(\mathbf{W}^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}^{(l)}\mathbf{d}_{t-1}^{(l+1)} + \mathbf{b}^{(l)}\right), \tag{3}$$

where $\mathbf{W}^{(l)}$ and $\mathbf{U}^{(l)}$ are weight matrices of the l^{th} layer, $\mathbf{b}^{(l)}$ is a bias vector, $\mathbf{d}_t^{(l)}$ and $\mathbf{d}_t^{(l+1)}$ represent the input and output for layer l at time t, respectively, $\mathbf{d}_{t-1}^{(l+1)}$ is the feedback from the previous step, $\mathcal{R}^{(l)}(\cdot)$ stands for the relation function for the input and output of the l^{th} RNN hidden layer, and the activation function often selects the *hyperbolic tangent* denoted by $\tanh$, which is $\delta_h(x) = (e^{2x}-1)/(e^{2x}+1)$. Using typical stochastic gradient descent method to train a recurrent network, the back-propagated error signals tend to zero that implies a prohibitively-long convergence time. To tackle this gradient-vanishing problem, Hochreiter and Schmidhuber proposed Long Short-Term Memory in their pioneer work of [22], which introduced *cell* and *gate* into the RNN structure. A typical LSTM cell has three gates: an *input* gate controlling the extent of new information flows into the cell, a *forget* gate to filter out useless memory, and an *output* gate that controls the extent to which the memory is applied to generate the activation. The upper part of Fig.3 shows the graphical depiction of a deep LSTM network consisting of an input layer, L hidden layers, and an output layer. Let's use the l^{th} hidden layer as an example to shed light on how an activation signal goes through the network. There are two hidden states - the short-term state $\mathbf{s}_{t-1}^{(l)}$ and the long-term state $\mathbf{c}_{t-1}^{(l)}$. The input $\mathbf{d}_t^{(l)}$ and $\mathbf{s}_{t-1}^{(l)}$ jointly activate four fully connected (FC) layers, generating the activation vectors for the gates, i.e.,

$$\begin{cases} \mathbf{i}_t^{(l)} = \delta_g\left(\mathbf{W}_i^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_i^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_i^{(l)}\right) \\ \mathbf{o}_t^{(l)} = \delta_g\left(\mathbf{W}_o^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_o^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_o^{(l)}\right), \\ \mathbf{f}_t^{(l)} = \delta_g\left(\mathbf{W}_f^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_f^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_f^{(l)}\right) \end{cases} \tag{4}$$

where $\mathbf{W}$ and $\mathbf{U}$ are weight matrices for the FC layers, $\mathbf{b}$ represents bias, the subscripts i, o, and f associate with the input, output, and forget gate, respectively, and δ_g stands for the logistic *Sigmoid* function $\delta_g(x) = 1/(1+e^{-x})$. The current long-term state $\mathbf{c}_t^{(l)}$ is obtained by first throwing away outdated memory at the forget gate and then adding new information selected by the input gate, i.e., $\mathbf{c}_t^{(l)} = \mathbf{f}_t^{(l)} \otimes \mathbf{c}_{t-1}^{(l)} + \mathbf{i}_t^{(l)} \otimes \mathbf{g}_t^{(l)}$, where the operator $\otimes$ denotes the Hadamard product (element-wise multiplication) and $\mathbf{g}_t^{(l)} = \delta_h(\mathbf{W}_g^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_g^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_g^{(l)})$. The output of this hidden layer is computed by

$$\mathbf{d}_t^{(l+1)} = \mathcal{L}^{(l)}\left(\mathbf{d}_t^{(l)}\right) = \mathbf{o}_t^{(l)} \otimes \delta_h\left(\mathbf{c}_t^{(l)}\right), \tag{5}$$

where $\mathcal{L}^{(l)}(\cdot)$ represents the input-output function for the l^{th} LSTM layer.

Despite of its short history, LSTM has achieved a great success and been commercially applied in many AI products such as Apple Siri and Google Translate. Since its emergence, the research community published a number of its variants, among which GRU proposed by Cho *et al.* in [23] draws lots of attention. It's a simplified version with fewer parameters,

but it exhibits even better performance over LSTM on certain smaller and less frequent datasets. To simplify the structure, a GRU memory cell has only a single hidden state, and the number of gates is reduced to two: the *update* and *reset* gate. The activation vector for the update gate is computed by $\mathbf{z}_t^{(l)} = \sigma_g(\mathbf{W}_z^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_z^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_z^{(l)})$, which decides the extend to which the memory content from the previous state will remain in the current state. The reset gate controls whether the previous state is ignored, and when it tends to 0, the hidden state is reset with the current input. It is given by $\mathbf{r}_t^{(l)} = \sigma_g(\mathbf{W}_r^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_r^{(l)}\mathbf{s}_{t-1}^{(l)} + \mathbf{b}_r^{(l)})$. Likewise, the previous hidden state $\mathbf{s}_{t-1}^{(l)}$ goes through the cell, drops outdated memory, and inserts some now content, generating the current hidden state, that is

$$\begin{aligned}\mathbf{s}_t^{(l)} &= (1-\mathbf{z}_t^{(l)}) \otimes \mathbf{s}_{t-1}^{(l)} \\ &\quad + \mathbf{z}_t^{(l)} \otimes \sigma_h\left(\mathbf{W}_s^{(l)}\mathbf{d}_t^{(l)} + \mathbf{U}_s^{(l)}(\mathbf{r}_t^{(l)} \otimes \mathbf{s}_{t-1}^{(l)}) + \mathbf{b}_s^{(l)}\right).\end{aligned} \tag{6}$$

The hidden state is also equal to its output of this hidden layer, i.e., $\mathbf{d}_t^{(l+1)} = \mathcal{G}^{(l)}(\mathbf{d}_t^{(l)}) = \mathbf{s}_t^{(l)}$, where $\mathcal{G}^{(l)}(\cdot)$ denotes the input-output function.

B. DL-based Channel Predictor

To shed light on the principle of a DL-based predictor, as shown in Fig.3, the chain of signal reception at the receiver is demonstrated. The predictor is inserted at the end of a channel estimator and generates predicted CSI to replace outdated CSI as the input for a relay selector. It is transparent and therefore an ORS system can be smoothly upgraded to a PRS system without any other modifications. In such a *distributed-selection* method, each relay requires to process only local CSI $h_{k,d}[t]$. As we know, a complex-valued fading coefficient can be expressed in polar form as $h_{k,d}[t] = a_{k,d}[t]e^{j\theta_{k,d}[t]}$, where $a_{k,d}[t]$ and $\theta_{k,d}[t]$ denote the magnitude and phase, respectively. Because the selection relies on the value of SNR, only the knowledge of magnitude $a_{k,d}[t]$ is enough, rather than complex-valued $h_{k,d}[t]$, which in turn can simplify the implementation of the channel predictor by employing a neural network with real-valued weights and biases. Feeding $a_{k,d}[t]$ into the input feed-forward layer obtains one-dimensional output $\mathbf{d}_t^{(1)} = d_t^{(1)} = \delta_h(w^{(i)}a_{k,d}[t]+b^{(i)})$, where $w^{(i)}$ and $b^{(i)}$ denote the weight and bias of the input layer. The activation of the 1^{st} hidden layer is exactly $\mathbf{d}_t^{(1)}$, then $\mathbf{d}_t^{(2)} = \mathcal{L}^{(1)}(\mathbf{d}_t^{(1)})$ is generated and forwarded to the 2^{nd} hidden layer, where $\mathcal{L}^{(1)}(\cdot)$ is defined in (5). The activation goes through the network until the output layer gets the predicted CSI, which is computed by $\breve{a}_{k,d}[t+1] = \delta_h(\mathbf{W}^{(o)}\mathbf{d}_t^{(L)} + b^{(o)})$, where $\mathbf{W}^{(o)}$ and $b^{(o)}$ denote the weight matrix and bias of the output layer, and the activation of the last hidden layer equals to $\mathbf{d}_t^{(L)} = \mathcal{L}^{(L)}(\ldots\mathcal{L}^{(2)}(\mathcal{L}^{(1)}(\mathbf{d}_t^{(1)})))$. The building of a deep recurrent network is flexible, for example, we can apply a hybrid network consisting of RNN, GRU, and LSTM layers, like $\mathbf{d}_t^{(L)} = \mathcal{G}^{(L)}(\ldots\mathcal{L}^{(2)}(\mathcal{R}^{(1)}(\mathbf{d}_t^{(1)})))$.

V. Computational Complexity

In the context of cooperative diversity, the computational complexity mainly arises from multi-relay coordination and synchronization [2]. The simplicity of ORS is achieved thanks to single-relay transmission that substantially lowers the amount of signalling overhead among multiple relays. A direct comparison of different schemes is not easy and does not provide real insight. That is why most of the works in this field [2]–[12] did not provide a quantitative analysis. On the other hand, the complexity of the proposed scheme comes mainly from the DL-based predictor, which is always a concern for the application of deep learning. From a practical perspective, it is more meaningful to make clear its demand on computing resources in comparison with the capability of COTS hardware. Hence, let's focus on assessing the complexity of the predictors in terms of floating-point operations per second (FLOPS).

A deep recurrent network can be quantitatively modelled as follows: an input layer with N_i neurons, an output layer with N_o neurons, and L hidden layers, which has N_h^l neurons at layer $l = 1,\ldots,L$. To begin with the input layer, it computes $\delta_h(\mathbf{W}^{(i)}\mathbf{d} + \mathbf{b}^{(i)})$, where the matrix multiplication generates $N_iN_h^1$ floating-point multiplicative operations and $(N_i-1)N_h^1$ additive operations, and the addition of the bias vector consumes N_h^1 operations, amounting to a total of $2N_iN_h^1$. Note that the amount of computation raised by the activation function is negligible compared to the matrix multiplication, which is usually ignored in the calculation of complexity for deep learning. Likewise, it is easy to know that the output layer corresponds to $2N_h^LN_o$. For an RNN hidden layer as given in (3), the number of operations equals to $O^l = (2N_h^{l-1}-1)N_h^l + (2N_h^l-1)N_h^l + N_h^l$, where the first term corresponds to the calculation of $\mathbf{W}^{(l)}\mathbf{d}_t^{(l)}$, the second is for $\mathbf{U}^{(l)}\mathbf{d}_{t-1}^{(l+1)}$, and the third is due to the addition of the bias. For simplicity, O^l can be approximated to $2N_h^{l-1}N_h^l + 2(N_h^l)^2$. Then, the overall complexity for a simple RNN is given by

$$O_{rnn} \approx 2\left[N_iN_h^1 + N_h^LN_o + \sum_{l=1}^{L}\left(N_h^{l-1}N_h^l + \left(N_h^l\right)^2\right)\right], \tag{7}$$

where we apply $N_h^0 = N_i$ for a simpler expression. As derived from (4)-(5), the number of operations for the matrix multiplication on an LSTM layer is 4 times that of an RNN layer, i.e., $4O^l$. The computation for the gate control, which has totally $7N_h^l - 3$ operations, can be neglected. Therefore, the complexity of an LSTM network is approximated by

$$O_{lstm} \approx 2\left[N_iN_h^1 + N_h^LN_o + \sum_{l=1}^{L}4\left(N_h^{l-1}N_h^l + \left(N_h^l\right)^2\right)\right]. \tag{8}$$

Similarly, we can derive the expression for GRU, i.e.,

$$O_{gru} \approx 2\left[N_iN_h^1 + N_h^LN_o + \sum_{l=1}^{L}3\left(N_h^{l-1}N_h^l + \left(N_h^l\right)^2\right)\right] \tag{9}$$

Note that the above expressions are the complexity per prediction step, we need to multiply (7)-(9) with the frequency of

prediction denoted by f_p, i.e., the number of steps performed per second, to figure out FLOPS.

Given the concrete values of these parameters, the complexity of the predictor is quantified to compare with the capacity of COTS computing hardware. Suppose the applied deep neural network has two LSTM hidden layers with $N_h^1 = N_h^2 = 25$ neurons[1]. The input for the predictor at the k^{th} relay is $a_{k,d}[t]$, corresponding to $N_i = N_o = 1$. It amounts to $O_{lstm} = 15,300$ floating-point operations *per prediction* in terms of (8). The interval of prediction step is assumed to be 1ms, the frequency of prediction equals to $f_p = 1,000$, resulting in 15.3MFLOPS. In comparison with off-the-shelf Digital Signal Processors (DSPs), e.g., TI C6678, which provides a computation capacity of up to 179GFLOPS, the required computing resource occupies less than 0.01% of a single DSP chip. Taking into account its back-compatibility to legacy hardware, we further check low-end DSPs. Given TI C6748 that has computation power of 2.7GFLOPS as an example, the resource required by the predictor is around 0.6%. In a nutshell, the complexity of the DL-based channel predictor applied for PRS is quite affordable, if not negligible.

VI. Simulation results

In this section, we clarify how to select the hyper-parameters of a deep recurrent network to obtain high prediction accuracy and then make use of Monte-Carlo simulations to evaluate the outage probability and channel capacity of PRS, compared with the existing schemes including ORS and OSTC. Following the channel assumption adopted by most of the previous works in this field, we would apply single-antenna flat-fading *i.i.d.* channels. Each channel follows the Rayleigh distribution with an average power gain of 0dB, where its fading coefficient $h \sim \mathcal{CN}(0,1)$. The default maximal Doppler frequency shift is set to f_d=100Hz, emulating fast fading environment. Continuous-time channel responses are sampled with a rate of f_s=1KHz, adhering to the assumption of flat fading, and therefore the interval of samples is T_s=1ms. Each channel generates a series of 10^6 consecutive samples $\{h[t]\,|t{=}1,2,\ldots,10^6\}$. As usual, an $\mathbb{EE}$ target rate of R=1bps/Hz is applied for outage calculation. The total transmit power P is equally allocated between two phases, where the source's power is P_s=$0.5P$, resulting in an average SNR $\bar{\gamma}_{s,k}$=$0.5P/\sigma_n^2$, while $\bar{\gamma}_{k,d}$=$0.5P/\sigma_n^2$ for the $\mathbb{RD}$ link.

A. Training the Predictor

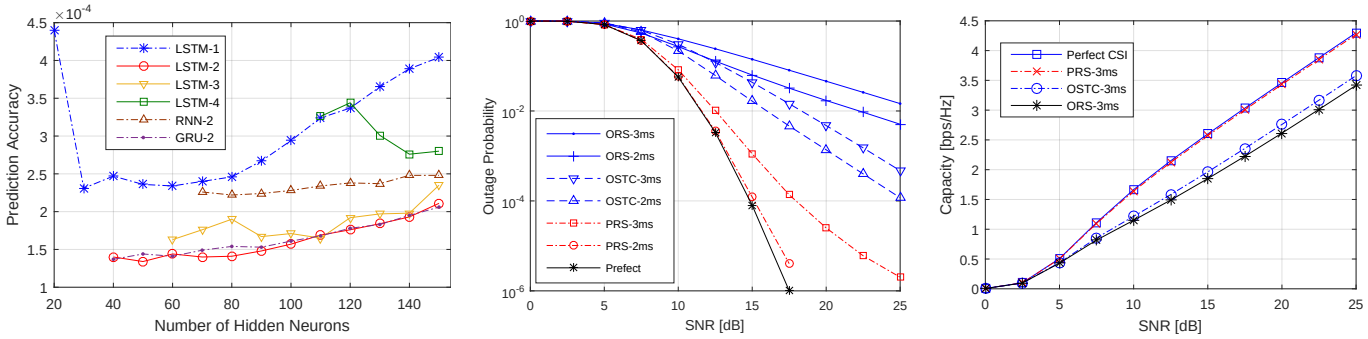
The hyper-parameters of a deep network such as the number of layers or neurons have a substantial impact on prediction accuracy. It is worth clarifying how to tune a deep network on demand. A training process starts from an initial state where all weights and biases are randomly selected. The input of the predictor at the relay is $a_{k,d}[t]$ and the output is its D-step-ahead prediction $\breve{a}_{k,d}[t{+}D]$. To measure prediction accuracy, the mean squared error (MSE) is applied as the cost function, namely $\text{MSE} = \frac{1}{T}\sum_{t=1}^{T} |a_{k,d}[t{+}D] - \breve{a}_{k,d}[t{+}D]|^2$, where T is the total number of channel samples for evaluation. Using the *batch* training, a batch of 256 samples is fed into the network per step. The output is compared with the desired values and the resultant error signals are propagated back through the network to update the weights by means of training algorithms such as the Adam optimizer used in our simulation. After 10 epochs, the trained network is employed to predict CSI.

Fig.4a compares the prediction accuracy of the predictors with different hyper-parameters. Let's first look at the impact of the number of layers and the number of neurons. Starting from an LSTM network with a single hidden layer, denoted by *LSTM-1* in the legend of the figure, its accuracy curve as a function of the number of hidden neurons likes an 'U' shape. That is because the network suffers from the *under-fitting* problem with only 20 neurons in the hidden layer, while the *over-fitting* problem appears using over 80 neurons. To make a fair comparison, the horizontal axis represents the total number of hidden neurons, which are evenly allocated across layers. For instance, the point of '60' in the horizontal axis means a 2-hidden-layer network with 30 neurons at either layer (denoted by *LSTM-2*), a 3-hidden-layer network with 20 neurons per layer (denoted by *LSTM-3*), or a single layer with 60 hidden neurons. No matter how many neurons in its single hidden layer, *LSTM-1* cannot reach the high accuracy achieved by *LSTM-2* and *LSTM-3*, justifying the benefit of deep learning. But it does not mean that the more layers, the better, as shown by the worse result of *LSTM-4*, which has 4 hidden layers. After known that 2-hidden-layer is the best choice for LSTM, we further observe the recurrent networks with 2 RNN or GRU hidden layers, indicated by *RNN-2* and *GRU-2*, respectively. As we can see, GRU performs as good as LSTM, whereas RNN is weak. As a result, we select a 2-hidden-layer LSTM network with 25 neurons at either layer, upon which the numerical results in the following figures are derived.

B. Performance Comparison

We further compare the outage performance of three relaying schemes in a cooperative network with K=8 relays, as illustrated in Fig.4b. The relay selection with the perfect knowledge of CSI (i.e., ρ=1) is used as the benchmark, which has the diversity order of 8 and decays at a rate of $1/\bar{\gamma}^8$, where $\bar{\gamma} = P/\sigma_n^2$ is the average $\mathbb{EE}$ SNR. With the delay of $\tau = 2$ and 3ms, the quality of outdated CSI drops to $\rho_o = J_0(0.4\pi) \approx 0.6425$ and $J_0(0.6\pi) \approx 0.2906$, respectively, which substantially deteriorates the performance. The diversity of ORS falls into 1, i.e., no diversity, and the curve decays slowly at a rate of $1/\bar{\gamma}$ in the high SNR regime. OSTC can redeem some loss and achieve the diversity order of 2 by using a pair of relays, but its gap to the benchmark is still large, more than 7dB at the level of 10^{-3}. Making use of channel prediction, the quality of CSI can be improved to $\rho > 0.95$. The proposed scheme achieves nearly the optimal performance with the horizon of 2ms (by setting $D = 2$ steps prediction), and remarkably outperforms OSTC with a gain of approximately 8dB in the case of 3ms. Moreover, the channel capacities for different schemes given $\tau = 3\text{ms}$ are

[1]The selection of such hyper-parameters will be justified in the next section.



(a) (b) (c)

Fig. 4. (a) Prediction accuracy in terms of the number of hidden neurons. (b) Comparison of outage probability for ORS, OSTC, and PRS in a cooperative network with K=8 relays; (c) Comparison of channel capacity for ORS, OSTC, and PRS in a cooperative network with K=8 relays.

comparatively illustrated in Fig.4c. At the SNR of $\bar{\gamma}$=20dB, for instance, ORS, OSTC, and PRS achieves 2.6, 2.75, and 3.5bps/Hz, respectively, where ORS suffers from a loss of around 1bps/Hz but PRS achieves a near-optimal capacity.

VII. Conclusions

In this paper, we proposed a deep-learning-aided cooperative diversity method for mobile terminals without an antenna array to cultivate the benefit of spatial diversity. A recurrent neural network was deliberately built to improve the timeliness of channel state information applied for selecting a single opportunistic relay. Simply inserting a channel predictor between the channel estimator and relay selector, an ORS system can be upgraded to a PRS system without any other modifications, making it transparent and easier to compatible with the existing systems and standards. It achieves the optimal performance with the full diversity order equaling to the number of cooperating relays in slow fading wireless environments, and substantially outperforms the existing schemes in fast fading channels. It inherits the simplicity of ORS by avoiding multi-relay coordination and synchronization, and the computational complexity arising from fading channel prediction is negligible compared with COTS hardware. From the perspective of *performance*, *compatibility*, and *complexity*, it is viewed as a good candidate for next-generation cooperative networks.

References

[1] W. Jiang, "Device-to-device based cooperative relaying for 5G network: A comparative review," *ZTE Commun.*, vol. 15, no. S1, pp. 60–66, Jun. 2017.

[2] A. A. Nasir *et al.*, "Timing and carrier synchronization with channel estimation in multi-relay cooperative networks," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 793–811, Feb. 2012.

[3] J. N. Laneman and G. W. Wornell, "Distributed space-time-coded protocols for exploiting cooperative diversity in wireless networks," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2415–2425, Oct. 2003.

[4] A. Bletsas *et al.*, "A simple cooperative diversity method based on network path selection," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 3, pp. 659–672, Mar. 2006.

[5] W. Jiang *et al.*, "An MGF-based performance analysis of opportunistic relay selection with outdated CSI," in *Proc. IEEE VTC'2014-Spring*, Seoul, South Korea, May 2014.

[6] J. L. Vicario *et al.*, "Opportunistic relay selection with outdated CSI: Outage probability and diversity analysis," *IEEE Trans. Wireless Commun.*, vol. 8, no. 6, pp. 2872–2876, Jun. 2009.

[7] W. Jiang *et al.*, "Opportunistic relaying over aerial-to-terrestrial and device-to-device radio channels," in *Proc. IEEE ICC'2014*, Sydney, Australia, Jul. 2014, pp. 206–211.

[8] L. Xiao and X. Dong, "Unified analysis of generalized selection combining with normalized threshold test per branch," *IEEE Trans. Wireless Commun.*, vol. 5, no. 8, pp. 2153–2163, Aug. 2006.

[9] Y. Li *et al.*, "On the design of relay selection strategies in regenerative cooperative networks with outdated CSI," *IEEE Trans. Wireless Commun.*, vol. 10, no. 9, pp. 3086–3097, Sep. 2011.

[10] W. Jiang, T. Kaiser, and A. J. H. Vinck, "A robust opportunistic relaying strategy for co-operative wireless communications," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2642–2655, Apr. 2016.

[11] W. Jiang, H. Cao, and T. Kaiser, "Opportunistic space-time coding to exploit cooperative diversity in fast-fading channels," in *Proc. IEEE ICC'2014*, Sydney, Australia, Jun. 2014, pp. 4814–4819.

[12] W. Jiang *et al.*, "Achieving high reliability in aerial-terrestrial networks: Opportunistic space-time coding," in *Proc. IEEE Eur. Conf. on Net. and Commun. (EUCNC)*, Bologne, Italy, Jun. 2014.

[13] W. Jiang and H. D. Schotten, "Neural network-based fading channel prediction: A comprehensive overview," *IEEE Access*, vol. 7, pp. 118 112–118 124, Aug. 2019.

[14] ——, "Deep learning for fading channel prediction," *IEEE Open J. Commun. Society*, vol. 1, pp. 320–332, Mar. 2020.

[15] W. Jiang and H. Schotten, "Neural network-based channel prediction and its performance in multi-antenna systems," in *Proc. IEEE Vehicular Tech. Conf. (VTC)*, Chicago, USA, Aug. 2018.

[16] W. Jiang and H. D. Schotten, "Recurrent neural network-based frequency-domain channel prediction for wideband communications," in *Proc. IEEE Vehicular Tech. Conf. (VTC)*, Kuala Lumpur, Malaysia, Apr. 2019.

[17] W. Jiang, H. Schotten, and J. Y. Xiang, "Neural networkbased wireless channel prediction," in *Machine Learning for Future Wireless Communications*, F. L. Luo, Ed. United Kindom: John Wiley&Sons and IEEE Press, 2019, ch. 16.

[18] W. Jiang and H. D. Schotten, "Multi-antenna fading channel prediction empowered by artificial intelligence," in *Proc. IEEE Veh. Tech. Conf. (VTC)*, Chicago, USA, Aug. 2018.

[19] W. Jiang and H. Schotten, "Recurrent neural networks with long short-term memory for fading channel prediction," in *Proc. IEEE Veh. Tech. Conf. (VTC)*, Antwerp, Belgium, May 2020.

[20] W. Jiang, M. Strufe, and H. Schotten, "Long-range MIMO channel prediction using recurrent neural networks," in *Proc. IEEE Consumer Commun. & Netw. Conf. (CCNC)*, Los Vegas, USA, Jan. 2020.

[21] W. Jiang and H. D. Schotten, "A deep learning method to predict fading channel in multi-antenna systems," in *Proc. IEEE Veh. Tech. Conf. (VTC)*, Antwerp, Belgium, May 2020.

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

[23] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *preprint arXiv:1406.1078*, Jun. 2014.