Deep Learning for Frame Error Prediction using a DARPA Spectrum Collaboration Challenge (SC2) Dataset

Xiwen Zhang, *Student Member, IEEE*, Abu Shafin Mohammad Mahdee Jameel, *Student Member, IEEE*, Ahmed P. Mohamed, *Student Member, IEEE*, Aly El Gamal, *Senior Member, IEEE*

***Abstract*—We demonstrate a first example for employing deep learning in predicting frame errors for a Collaborative Intelligent Radio Network (CIRN) using a dataset collected during participation in the final scrimmages of the DARPA SC2 challenge. Four scenarios are considered based on randomizing or fixing the strategy for bandwidth and channel allocation, and either training and testing with different links or using a pilot phase for each link to train the deep neural network. Interestingly, we unveil the efficacy of randomization in improving detection accuracy and the generalization capability of certain deep neural network architectures with Bootstrap Aggregating (Bagging).**

I. Introduction

WIRELESS networks are undergoing major transformations as they face major challenges that range from supporting an unprecedented scale of the number of users to meeting the Quality of Service (QoS) requirements of new applications like Virtual Reality (VR), intelligent transportation systems, and the Internet of Things (IoT). At the same time, computational advances - most notably in Deep Learning (DL) algorithms and supporting hardware - carry the promise of enabling new possibilities, that were not considered feasible before, such as a completely autonomous and agile wireless network that can efficiently co-exist in a complex and dynamic environment that is affected by several other independently designed networks. Three characteristics of modern wireless communication make DL an attractive option. Firstly, modeling next generation complex and dynamic networks is difficult, and depends on assumptions which may not always hold true [1]. Secondly, massive amount of data can be easily collected within a small time resolution, making deep network training easier [2]. Thirdly, many of the basic operations and objectives of state of the art deep neural network architectures, like convolution (or mathematical correlation) and capturing temporal correlations - as in Long Short Term (LSTM) cells - have been also central for the design of wireless communication systems. Thus, DL based methods are strong candidates for modern wireless communication systems [3], [4].

The authors are with the School of Electrical and Computer Engineering of Purdue University, West Lafayette, IN (e-mail: {zhan2977, amahdeej, mohame23, elgamala}@purdue.edu).

This work is supported by DARPA and AFRL under grant no. 108818. The views, opinions, and/or findings expressed are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

Approved for Public Release, Distribution Unlimited.

The application of DL in the physical wireless communication layer has recently witnessed intense research focus, with efforts in the fields of estimation and detection [5], [6], encoding and decoding [7], scheduling and power allocation [8], [9], modeling and identification [10], [11] of wireless channels, as well as end to end optimization [12], [13], modulation recognition [14], [15], and interference identification [16].

One major improvement in the recently introduced fifth generation standard (5G) is the possibility of spectrum sharing between 4G and 5G networks [17], [18]. Traditional approaches to spectrum sharing are severely limited, as initial rule based sharing approaches are not receptive to dynamic adjustments [19]. Further, game theory based approaches to spectrum sharing [20] often rely heavily on unrealistic assumptions. Recently, with the move towards software defined radios, approaches incorporating machine learning based techniques for route allocation have gained attention [9], [21]–[23].

The three year DARPA Spectrum Collaboration Challenge (SC2), which concluded in 2019, was intended to facilitate the research to automate the labor-intensive inefficient process of spectrum management [24]. Research teams competing in the challenge have heterogeneously developed intelligent radio networks that autonomously coexist with significant independence, increasing the common throughput significantly [25]. Teams employed hand tuned expert systems, as well as machine learning algorithms to solve the problem of dynamic spectrum sharing [26]. The data gathered by various SC2 teams have been used to investigate novel techniques, some of them incorporating DL. In [27], a pre-trained DL network from the computer vision domain was demonstrated to achieve good collision detection performance.

For environments like those emulated in SC2 that depend on collaborative spectrum utilization schemes, optimization of the transmission parameters in a software defined radio network is vital [28]. State based modeling like the Markov Trace Analysis [29], Hidden Markov Model [30], and pairwise error probability model [31] predict future frame errors through the state of the current and past frames, but do not take into account properties of the physical channel. Channel measurement based models incorporate channel properties like path loss, fading, and signal reception [32]. Recently, in [33], a frame error probability prediction algorithm in a bit-interleaved coded modulation orthogonal frequency division multiplexing (BICM-OFDM) system was proposed. Although the authors also employ a deep learning approach and achieve

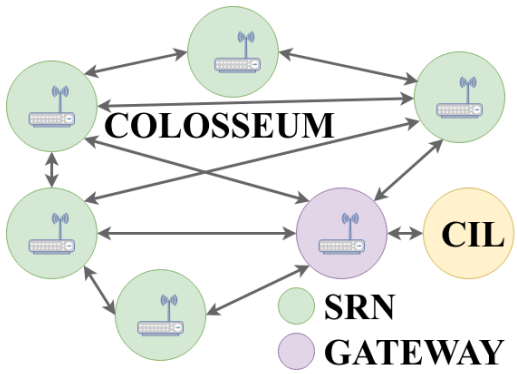

Fig. 1: A DARPA SC2 Collaborative Intelligent Radio Network (CIRN) that consists of a Gateway node and multiple Software Radio Nodes (SRN). The wireless environment is emulated through DARPA's Colosseum, and different CIRNs communicate through the CIRN Interaction Language (CIL).

promising results, their experiments are based on numerical simulations. Here instead, by using the data collected in SC2, which rely on DARPA's Colosseum emulator [34], we demonstrate the feasibility of deep learning based frame error prediction in a more realistic scenario.

II. Dataset

The dataset is collected from the third and final Phase of SC2. For every match, five teams are selected from all qualified teams, and each team runs an independently designed Collaborative Intelligent Radio Network (CIRN) of 10 radio nodes. Each match is divided into stages where each network is given a number of individual mandates (IM) at each stage. Every IM specifies an IP traffic flow to be delivered between source and destination IP addresses within the CIRN. Further, it specifies the required QoS requirements, such as maximum allowed latency and minimum allowed throughput, as well as an associated number of reward points to be granted upon successful completion. During a match, the objective for each team is to maximize its own accumulated score, while keep other teams above pre-specified thresholds. The latter enforces collaboration between teams, because no team can win simply by being greedy. Additionally, all five CIRNs can communicate over a specified data link using a common protocol, called the CIRN Interaction Language (CIL).

As a competitor in DARPA SC2, in every match that we participated in during the last two scrimmages preceding the final event - namely Scrimmages 4 and 5 -, we constantly collected data from the 10 nodes in our CIRN. We organize the collected data into an SQLite database file, which can easily be viewed, accessed and manipulated using a wide variety of tools. In total, there are 53 tables in a database file, recording all the information we collect from 10 nodes throughout a match. Here, instead of a comprehensive documentation, we only introduce the fields in this dataset that are most relevant to the considered task of frame error prediction.

- Power Spectral Density (PSD): We record PSD measurements through two different sources. Every 100 ms, each node in our CIRN actively measures and records the observed PSD. In addition, peer CIRNs report their current spectrum usage, GPS information and transmit power through CIL. Based on received CIL messages, we calculate the estimated PSD for each radio node in our network.
- Channel Allocation: We record every channel allocation update event in a match, so that for every frame transmitted, we know the allocated channel for that transmitter at the moment of transmission.
- Modulation and Coding Scheme (MCS): We record the MCS selected for each frame exchanged in our CIRN.
- Transmit Power: For every frame transmitted, we record the transmit power.
- Noise Variance: For every frame, we measure and record the noise variance along the supporting radio link.
- Frame Decoding Error: For every frame detected by the receiver, we record whether it has been successfully decoded, which is also the target label we try to predict.

III. Deep Learning for Frame Error Prediction

As a first example using the SC2 dataset that we collected, we carry out an empirical investigation in this section on the feasibility and potential of using deep learning for predicting frame errors. We believe that such a capability can serve as a critical component for an intelligent radio network to autonomously select the optimal transmission parameters for each radio link, e.g., MCS, transmit power and channel allocation, as well as to determine the optimal flow scheduling strategy to meet QoS requirements.

A. Problem Setup

We naturally cast the task of frame error prediction as a binary classification problem. For each frame, the goal is to predict whether a frame will be decoded successfully before transmission. To this end, information relevant to a transmitted frame is fed to a neural network, including

- PSD measured by the receiver node right before the frame is transmitted.
- Noise variance of supporting radio link.
- MCS used for the considered frame.
- Transmit power used for the considered frame.
- Allocated channel for supporting radio link.

Given these information, a deep neural network is employed to predict whether this frame will be decoded successfully at the receiver's side. It is important to note that all input information must be measured or collected before the frame is actually transmitted. The neural network can provide useful information to the radio network only if it can predict frame decoding errors before frames are transmitted.

B. Data Preparation

We use two datasets collected from Scrimmages 4 and 5 of DARPA SC2 Phase 3, respectively.

Scrimmage 4 contains 35 matches. From them, we choose the 25 matches whose scenario bandwidth equals 20 MHz.

There are 284 radio links in these 25 matches, each of which contains 10,000-30,000 transmitted frames. We randomly pick 190 links for training and the other 94 for testing. In other words, **data corresponding to the radio links used in test time are never seen during training**. There are 6,522,245 frames in total. 67.1% of them have been decoded successfully on the receiver side, whereas errors happened while decoding the other 32.9% of frames.

Scrimmage 5 contains 100 matches in total, and 59 of them have a bandwidth of 20 MHz. These 59 matches contain 627 radio links. We use 418 of them for training and the other 209 for testing. Scrimmage 5 dataset contains 11,669,546 frames. 72.6% of them have been decoded successfully, while 27.4% have had decoding errors. Note that the ratio of training to testing links is roughly 2:1 for both scrimmages.

C. Neural Network Architectures and Training

In order to find a baseline performance with a traditional pattern recognition technique, a 3-Nearest Neighbor algorithm based on Euclidean distance to training examples was employed first, which achieved an accuracy of 56.06% and a weighted accuracy of 52.09% on Scrimmage 4 dataset, which is close to the performance of random guessing. This indicated the large scope for improvement potentially available through deep learning algorithms.

The input and output layers of a neural network are determined by the problem setup, and hence, for our problem, there are 20 units in the input layer and two units at the output layer, corresponding to positive (decoding error) and negative (successful decoding) labels. For the sake of obtaining interpretable results, we have chosen log softmax as the activation function in the output layer, and negative log likelihood (cross entropy) as the loss function (for a justification of these choices, please see [35, Chapter 6]).

We started by employing a very simple deep neural network as a baseline; the fully connected multi-layer perceptron (MLP). This MLP has four hidden layers with 100 units each, and a rectified linear unit (ReLU) is used as the activation function for the input and hidden layers. ReLU has the advantageous property of quick convergence while increasing the chance of avoiding gradient saturation.

We then experimented with Convolutional Neural Network (CNN) architectures. To summarize the justification for this choice of study, we note that CNN architectures present a very effective option for parameter sharing, and are expected to work well with high-dimensional data, where the dimensions correspond to similar data types (for more details, please see [35, Chapter 9]). We tested two CNN architectures, one with two hidden layers (CNN2), and the other with four hidden layers (CNN4). Each convolutional layer contains 16 feature maps. To facilitate stable optimization, we used 1D batch normalization after each convolutional layer. ReLU is employed as the activation function in hidden layers, and softmax in the output layer. Before the output layer, there is a 320×2 dense layer with a dropout rate of 0.5 for regularization.

While CNNs are good at capturing small scale correlations between neighboring input features from the same frame, they do not capture long term temporal correlations. However, when analyzing received wireless signals, it is natural to expect significant long term temporal correlations, as a present frame often strongly relies on past frames. Hence, a Recurrent Neural Networks (RNN) is expected to perform well. We tested two RNN networks, the first one with a single Long Short Term Memory (LSTM) layer (LSTM1), and the second with two LSTM layers (LSTM2). Each LSTM layer contains 100 LSTM units, so before the output layer, a 100×2 dense layer is employed. We note that RNNs in general are often slower due to difficulty of parallelizing training optimization (see [35, Chapter 10]).

As deep neural networks are overparameterized, there is a natural risk of overfitting. Further, as our goal is to detect the relatively rare events of frame errors, this risk is increased. We hence experimented with training using the following variations of the optimization algorithm:

- Weighted Loss Function (WLF): Here, we assign more weight to examples with frame errors such that the total weight assigned to frame errors is the same as that of successful frames.
- Bootstrap Aggregating (Bagging): We explored the potential of bagging (BG) using nine identical models (see [35, Chapter 7] for more details).

For training of all considered architectures, we used the Adam optimizer with a learning rate of .001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Batch size was fixed at 1024, and early stopping is adopted to decide the number of training epochs.

We further employed an alternative method for the train-test split, that we label in the figures as *(With Pilot)*, which represents a scenario where a fraction of $\frac{2}{3}$ of the frames for each link is used as training data and the remaining as testing data. Our objective here is to quantify the negative impact on generalization performance due to testing with links that were never seen during training. The $\frac{2}{3}$ fraction was chosen to be similar to the fraction of links used for training in the original method. We note however that such a large fraction could be excessive in a practical scenario.

D. Results

We present the accuracy - as percentage - of the frame error prediction algorithms in Figure 2. All the architectures and training algorithm combinations mentioned in the previous subsection are represented here. Four experiments are performed for each combination, corresponding to binary choices of one of the two datasets and setting the training pilot option. We represent the results for the original method as solid lines, and the results for training with pilots as dashed lines. **Stars are used to mark the architecture and training algorithm combination that achieves the highest accuracy** for each of the four experiments.

Due to the unbalanced nature of the data, the accuracy measure does not convey complete information, as it is dominated by the majority class. We hence also present the weighted accuracy for all cases in Figure 3. The weighted accuracy gives larger weight to examples with frame errors such that there is

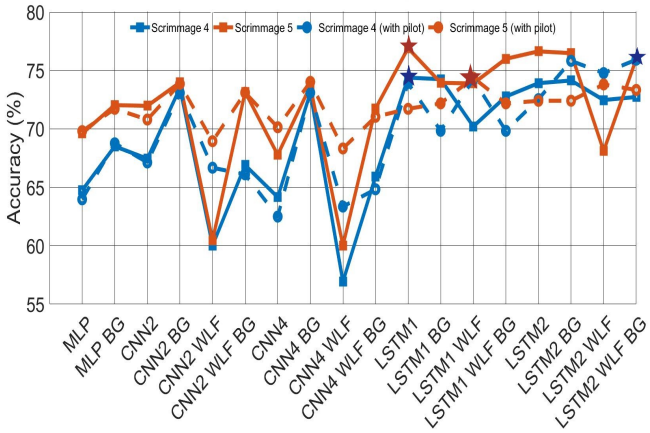

Fig. 2: Accuracy of different architectures for Frame Error Prediction, with and without a pilot training phase.

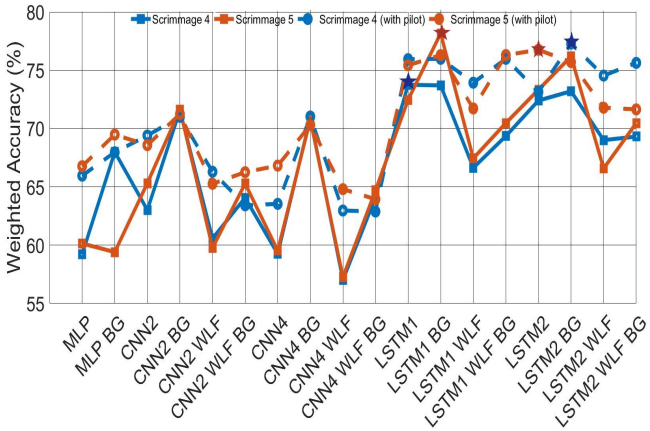

Fig. 3: Weighted Accuracy of different architectures for Frame Error Prediction, with and without a pilot training phase.

equal total weight assigned to both types of examples. It helps us better understand the anomaly detection performance.

We have the following observations from the results in Figure 2 and Figure 3.

1) Randomization for Data Collection: Even though randomizing the strategy for bandwidth and channel allocation may not be justified, and can be harmful, for real-time performance, it can be utilized to unveil structures in rare events through creating a diverse set of scenarios. We validate this by observing that even though the accuracy for Scrimmage 5 (fixed allocation strategy) tend to be consistently higher than that of Scrimmage 4 (randomized allocation strategy) as observed in Figure 2, the weighted accuracy values for Scrimmage 4 tend to be similar or slightly better as observed in Figure 3.
2) Effect of Pilot Phase: The positive impact on the best obtained accuracy due to training with a pilot phase is more evident for Scrimmage 4. We believe that this is because the link conditions can vary more significantly for a randomized channel allocation strategy, and hence adding data for the same link we are testing with can significantly improve training.
3) Best Architectures: All the best performing architectures are LSTM based. One can also observe that the LSTM-based architectures (right hand side half in the figures) typically deliver higher performance. We believe that this superiority of RNN architectures demonstrates the need for capturing long term temporal correlations across different frames.
4) Effect of Depth: We see little performance improvement as we increase the depth for the same architecture type.
5) Effect of Bagging: We obtain consistent improvement in almost all cases when employing Bagging, both in the case of regular and weighted loss functions, considered in terms of both regular and weighted accuracy.
6) Effect of Weighted Loss Function: We observe little or no improvement in performance due to assigning larger weights to erroneous frames during training.

It is to be noted that MLP training is not very stable, and the accuracy varies in repeated experiments. Here, a representative accuracy for MLP is given. This problem is not present for other architectures, where the results are easily reproducible.

IV. Regularization Techniques

To assess the potential of regularization, we utilized three different approaches. For brevity, only the accuracy of CNN2 on Scrimmage 4 is discussed (baseline is 67.44%).

1) K-NN on Output: Here, we set the new label for an example by taking a majority vote among original outcomes corresponding to the three nearest neighbors among testing examples (including itself). The accuracy increased to 72.09%.
2) Label Smoothing: Here, 5% of the training labels were randomly selected, and flipped before training, which gave an accuracy of 69.29%.
3) Input Noise Injection: We introduced additive random Gaussian noise to the input training examples. This resulted in an accuracy of 68.55%.

V. Future Research Directions

We presented a first example for employing deep learning for frame error prediction based on our recently collected SC2 dataset. For future work, we plan to perform a thorough experimental study to justify the need for employing deep learning for the considered task. Further, other RNN architectures such as Gated Recurrent Units (GRU) and the Convolutional Long Short Term Memory Deep Neural Network (CLDNN) will be investigated. Also, the effect of soft label smoothing through softmax units will be considered (see [35, Chapter 7] for more details). Finally, the effect of depth for the same architecture type will be further investigated.

Acknowledgment

The authors would like to thank all members of the DARPA SC2 team, and the competitor BAM! Wireless team, for fruitful discussions and valuable help with dataset collection.

References

[1] R. Burchfield, E. Nourbakhsh, J. Dix, K. Sahu, S. Venkatesan, and R. Prakash, "RF in the jungle: Effect of environment assumptions on wireless experiment repeatability," in *2009 IEEE International Conference on Communications*, 2009, pp. 1–6.

[2] M. S. Hadi, A. Q. Lawey, T. E. El-Gorashi, and J. M. Elmirghani, "Big data analytics for wireless and wired network design: A survey," *Computer Networks*, vol. 132, pp. 180–199, 2018.

[3] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Communications*, vol. 14, no. 11, pp. 92–111, 2017.

[4] H. Huang, S. Guo, G. Gui, Z. Yang, J. Zhang, H. Sari, and F. Adachi, "Deep learning for physical-layer 5G wireless techniques: Opportunities, challenges and solutions," *IEEE Wireless Communications*, 2019.

[5] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 1–5.

[6] D. Neumann, T. Wiese, and W. Utschick, "Learning the MMSE channel estimator," *IEEE Transactions on Signal Processing*, vol. 66, no. 11, pp. 2905–2917, 2018.

[7] H. Ye, L. Liang, G. Y. Li, and B.-H. F. Juang, "Deep learning based end-to-end wireless communication systems with conditional GAN as unknown channel," *arXiv preprint arXiv:1903.02551*, 2019.

[8] Z. Han and K. R. Liu, *Resource allocation for wireless networks: Basics, techniques, and applications*. Cambridge university press, 2008.

[9] K. I. Ahmed, H. Tabassum, and E. Hossain, "Deep learning for radio resource allocation in multi-cell networks," *IEEE Network*, vol. 33, no. 6, pp. 188–195, 2019.

[10] J. Wang, J. Tang, Z. Xu, Y. Wang, G. Xue, X. Zhang, and D. Yang, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2017, pp. 1–9.

[11] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433–445, 2018.

[12] R. Li, Z. Zhao, X. Zhou, G. Ding, Y. Chen, Z. Wang, and H. Zhang, "Intelligent 5G: When cellular networks meet artificial intelligence," *IEEE Wireless Communications*, vol. 24, no. 5, pp. 175–183, 2017.

[13] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2017.

[14] A. Fehske, J. Gaeddert, and J. H. Reed, "A new approach to signal classification using spectral correlation and neural networks," in *First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN)*, 2005, pp. 144–150.

[15] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *International conference on engineering applications of neural networks*. Springer, 2016, pp. 213–226.

[16] X. Zhang, T. Seyfi, S. Ju, S. Ramjee, A. El Gamal, and Y. C. Eldar, "Deep learning for interference identification: Band, training SNR, and sample selection," in *IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2019, pp. 1–5.

[17] R. Omana, "Using spectrum sharing to deploy 4G/5G capable wireless networks," Ericsson Inc., NCTA Technical Papers, Tech. Rep., October 2019.

[18] L. Wan, Z. Guo, Y. Wu, W. Bi, J. Yuan, M. Elkashlan, and L. Hanzo, "4G/5G spectrum sharing: Efficient 5G deployment to serve enhanced mobile broadband and internet of things applications," *IEEE vehicular technology magazine*, vol. 13, no. 4, pp. 28–39, 2018.

[19] D. Raychaudhuri and X. Jing, "A spectrum etiquette protocol for efficient coordination of radio devices in unlicensed bands," in *14th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC).*, vol. 1, 2003, pp. 172–176.

[20] R. Etkin, A. Parekh, and D. Tse, "Spectrum sharing for unlicensed bands," *IEEE Journal on selected areas in communications*, vol. 25, no. 3, pp. 517–528, 2007.

[21] L. A. DaSilva and I. Moerman, "Using deep learning and radio virtualisation for efficient spectrum sharing among coexisting networks," in *13th EAI International Conference on Cognitive Radio Oriented Wireless Networks (CROWNCOM)*, vol. 261. Springer, 2019.

[22] S. Giannoulis, C. Donato, R. Mennes, F. A. de Figueiredo, I. Jabandžić, Y. De Bock, M. Camelo, J. Struye, P. Maddala, M. Mehari *et al.*, "Dynamic and collaborative spectrum sharing: The SCATTER approach," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019.

[23] S. Jacob, V. G. Menon, S. Joseph, P. Vinoj, A. Jolfaei, J. Lukose, and G. Raja, "A novel spectrum sharing scheme using dynamic Long Short-Term Memory with CP-OFDMA in 5G networks," *IEEE Transactions on Cognitive Communications and Networking*, 2020.

[24] DARPA. Spectrum Collaboration Challenge (SC 2). (Date accessed Mar 02 2020). [Online]. Available: https://www.spectrumcollaborationchallenge.com

[25] D. Stojadinovic, F. A. de Figueiredo, P. Maddala, I. Seskar, and W. Trappe, "SC2 CIL: Evaluating the spectrum voxel announcement benefits," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019.

[26] C. Bowyer, D. Greene, T. Ward, M. Menendez, J. Shea, and T. Wong, "Reinforcement learning for mixed cooperative/competitive dynamic spectrum access," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2019.

[27] H. N. Nguyen, T. Vo-Huu, T. Vo-Huu, and G. Noubir, "Towards adversarial and unintentional collisions detection using deep learning," in *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*, 2019, pp. 22–24.

[28] A. Stamou, G. Kakkavas, K. Tsitseklis, V. Karyotis, and S. Papavassiliou, "Autonomic network management and cross-layer optimization in software defined radio environments," *Future Internet*, vol. 11, no. 2, p. 37, 2019.

[29] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A Markov-based channel model algorithm for wireless networks," *Wireless Networks*, vol. 9, no. 3, pp. 189–199, 2003.

[30] P. Ji, B. Liu, D. Towsley, and J. Kurose, "Modeling frame-level errors in GSM wireless channels," in *IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 3, 2002, pp. 2483–2487.

[31] N. Suga, N. Egashira, K. Yano, and T. Kumagai, "Frame error rate prediction for WLAN systems with multiband simultaneous transmission," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1–6.

[32] P. Barsocchi, G. Oligeri, and F. Potortì, "Measurement-based frame error model for simulating outdoor Wi-Fi networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1154–1158, 2009.

[33] V. Saxena, J. Jaldén, M. Bengtsson, and H. Tullberg, "Deep learning for frame error probability prediction in BICM-OFDM systems," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 6658–6662.

[34] DARPA. Colosseum Radio-Frequency Emulation Testbed. (Date accessed Mar 02 2020). [Online]. Available: https://www.spectrumcollaborationchallenge.com/the-colosseum/

[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.