IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 2, 2022, accepted March 18, 2022, date of publication March 28, 2022, date of current version April 5, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3162829

Confidence Aware Deep Learning Driven Wireless Resource Allocation in Shared Spectrum Bands

CHANAKA GANEWATTHA, ZAHEER KHAN, MATTI LATVA-AHO, (Senior Member, IEEE), AND JANNE J. LEHTOMÄKI, (Member, IEEE)
Centre for Wireless Communications (CWC), University of Oulu, 90014 Oulu, Finland
Corresponding author: Chanaka Ganewattha (chanaka.ganewattha@oulu.fi)

This work was supported by the Academy of Finland through the 6Genesis Flagship under Grant 318927.

ABSTRACT Deep learning (DL) driven proactive resource allocation (RA) is a promising approach for the efficient management of network resources. However, DL models typically have a limitation that they do not capture the uncertainty due to the arrival of new unseen samples with a distribution different than the data distribution available at DL model-training time, leading to wrong resource usage predictions. To address this, we propose a confidence aware DL solution for the robust and reliable predictions of wireless channel utilization (CU) in shared spectrum bands. We utilize an encoder-decoder based Bayesian DL model to generate prediction intervals which capture the uncertainties in wireless CU. We use the CU predictions to design a novel metric score which in turn is utilized to make an adaptive RA algorithm. We show that a DL model capturing uncertainty in CU can achieve higher data rates for a wireless network. Both DL driven predictions and RA models are tested using synthetic data as well as real CU data collected in the University of Oulu. Using analytical and simulations results, we also study the stability of the proposed RA algorithm and show that it converges to a Nash equilibrium (NE). Our results reveal that the proposed algorithm converges to an NE under $2N$ iterations where N is the number of network access points.

INDEX TERMS 6G, Bayesian neural networks, channel utilization, distribution change detection, dynamic wireless networks, game theory, predictive uncertainty, proactive resource allocation, shared spectrum bands.

I. INTRODUCTION

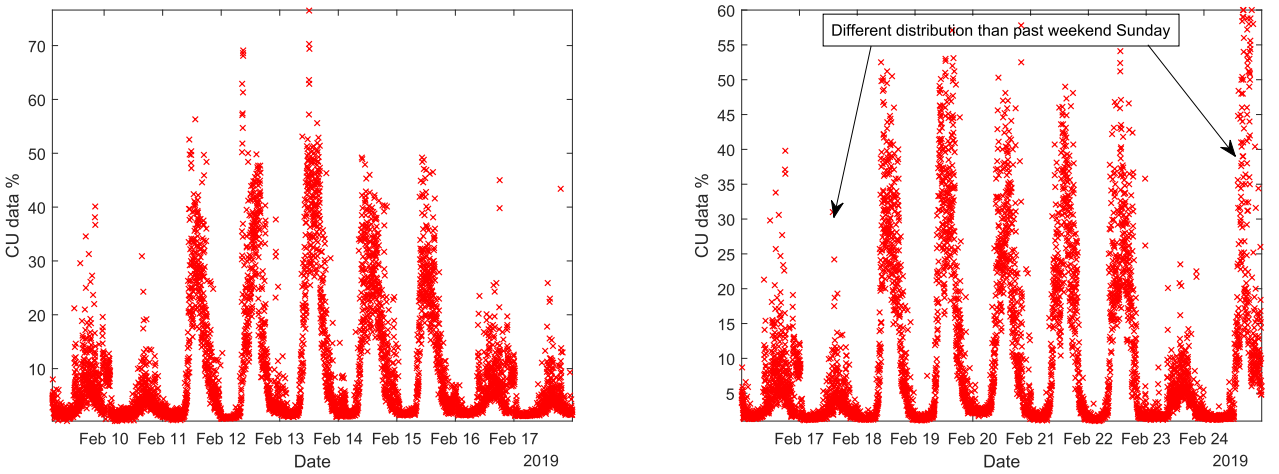
A. BACKGROUND AND MOTIVATION

For the sixth-generation (6G) of wireless networks, the prediction of their resource utilization variations using sophisticated deep learning (DL)-driven techniques can enable the network to proactively schedule resources for those services/network elements which have higher resource demands [1]–[3]. This approach is opposed to the reactive resource allocation approaches mainly adopted by existing networks which allocate resources based on current resource requirements. A proactive resource allocation solution which optimizes the resources beforehand is able to cater to the demands better and effectively avoid congestion of resources.

Efficient and robust management of network resources is a critical aspect for the success of the next generation of wireless networks [4]. Judicious resource allocation policies in wireless communication networks can guarantee the required quality of service (QoS), and can also maximize a wireless operator's revenue through optimal and efficient operation of its network [5]. Recently, cloud management of wireless networks in both licensed and unlicensed/shared spectrum bands have attracted the attention of both the industry and the research community. This interest is due to the cloud's ability to optimize and manage resources of an entire network with enhanced computation and analytics capabilities [6], [7]. To make the most out of a cloud managed network, tracking the right metrics using descriptive analytics and combining them with DL models for realistic predictions are significant for their efficient design. For example, the works in [8]–[10] used DL based predictions for network resource allocation.

Most of the existing studies using DL methods for proactive resource allocation in wireless networks such as [8], [9] have implicitly assumed that training and target datasets have the same distribution. Hence, they produce erroneous resource usage predictions when the target datasets have different distribution than the training dataset. This can lead to unreliable predictions as the changes in users' resource utilization in a network over time give rise to uncertainty in the DL models. Wireless datasets can be dynamic which

The associate editor coordinating the review of this manuscript and approving it for publication was Nagarajan Raghavan.



(a) CU data which has same CU distribution during weekends

(b) CU data which has a different CU distribution during weekends

FIGURE 1. Collected CU data showing distribution change during the weekend.

means distribution of these datasets can change from time to time. In the context of DL, the uncertainty arising due to the change in the data distribution from training to testing is called model misspecification [11].

To provide an example of model misspecification in wireless networks, in Fig. 1(a) and 1(b), we illustrate wireless channel utilization (CU)[1] collected by us over an unlicensed shared channel in the University of Oulu. The details of the data collection process is given in section III-A. CU is given as a percentage value between 0% and 100% and it represents the amount of wireless channel usage by various users and access points (Aps) within the measuring time interval t. Fig. 1(a) shows the collected CU data values in percentage for nine days. It can be seen from the figure that there is a daily pattern for weekdays (Feb. 11-15) and weekends (Feb. 10-11 and Feb. 16-17). Fig. 1(b) which presents the CU data for the next nine days shows that while there is still a daily pattern for weekdays, however, the weekend pattern does not hold due to high CU on Sunday, Feb. 24. A DL model may not be able to predict this kind of behavior as it might not have seen this behavior before in the training data. Other uncertainties associated with DL models are model uncertainty [11] which arises due to missing training data covering certain areas of the input domain and inherent noise [11] which arises due to uncertainty in the data generation function. Capturing these kinds of uncertainties associated with DL models are crucial for the correct operation of proactive resource allocation systems which rely on them.

Handling uncertainties using deep neural networks (DNNs) are problematic due to their inherent design. DNNs have a tendency to overfitting which can unfavorably impact their generalization capabilities. Moreover, they are overconfident about their predictions even for out-of-training distribution data [12]. Therefore, it is difficult to employ DNNs in estimating uncertainties. Bayesian Neural Networks (BNNs) on the other hand produce predictions by the aggregation of predictions from a large set of independent and average-performing predictors [12]. This can allow BNNs to make predictions better than DNNs and also enable them to estimate uncertainties in a meaningful way. This has motivated us to use a DL model based on BNN in our design.

[1]The CU data used in this work is pre-COVID period as due to COVID situation either the university has been closed or still both remote education and remote work are continued by many people leading to unusually low CU levels.

B. MAIN CONTRIBUTIONS

In this paper, we first focus on the design of a DL model that can perform robust and reliable predictions of wireless CU for proactive resource allocation in unlicensed shared spectrum bands. Our DL model also incorporates uncertainty estimation of CU predictions that is utilized for real-time change detection in CU data distribution. Our DL model is based on encoder-decoder framework of [11] which uses BNN to handle uncertainty in models. We use Algorithm 1 of [11] as an application in improving the predictions in wireless network resource utilization. We also use it to design a more efficient resource allocation algorithm which is not only proactive, but also reactive to the generated alarms via Algorithm 1 for resource utilization.

The main contributions of this paper can be summarized as:

1) The proposed DL model addresses the problem of model uncertainty and model misspecification. The DL model uncertainty which arises due to insufficient training samples is reduced by collecting a large number of real CU samples using our FPGA based radio frequency (RF) data processing module (see [13] for details of its implementation). Moreover, the DL model misspecification is quantified by using an encoder-decoder model on real wireless CU data. Rather than making only the point predictions, the

DL model also estimates the degree of uncertainty in predicted wireless CU values via the use of prediction intervals (Pis). A PI is a type of confidence interval used with predictions; it is a range of values that predicts the value of a new observation, based on an existing model.

2) We present a methodology for characterizing and measuring the robustness of the proposed prediction method and show that our DL model outperforms other models in terms of robustness. We also evaluate the prediction performance of the developed model using real CU data in terms of mean average percentage error (MAPE) and compare its performance with a baseline method (I), a long short term memory (LSTM) model, and a gated recurrent unit (GRU) model.
3) We utilize the DL prediction values for a given time period to design novel resource metric scores which in turn are used to find an automated, stable and efficient channel allocation solution for multiple wireless APs. In particular, the uncertainty estimate from the prediction model is used by the channel allocation algorithm to adapt the allocation decisions when the real observed CU values fall outside of the defined PI. Our results show that taking into account the uncertainty estimate from the prediction model can improve the channel allocation performance as compared to when no uncertainty estimate is utilized.
4) We evaluate the performance of the proposed algorithm in terms of average sum metric and average sum rate which reflect the effectiveness of all the APs for a given channel allocation. To evaluate the stability of the proposed algorithm under various scenarios, along with simulations, we also focus on analytical game theoretic concept of stability called Nash Equilibrium (NE). An (pure) NE represents an individually agreeable, or stable, allocation for scenarios where no wireless AP has an incentive to unilaterally deviate from the proposed resource allocation solution. We also show that the proposed algorithm requires no more than $2N$ resource allocation steps to stabilize, where N is the number of APs.

Rest of the paper is organized as follows. The next section presents related work including the application of DL in wireless networks. In Section III, we present the system model. In Section IV, we present the theoretical basis and the implementation details of the proposed DL model including the prediction performance results. Section V presents the novel metric scores and the proposed channel allocation algorithm. In Section VI, we evaluate the proposed algorithm under several different scenarios and discuss the simulation results. Section VII concludes the paper and also provides the future research directions.

II. RELATED WORK

DL is a subset of recent machine learning methods which is based on artificial neural networks (NNs). These NNs learn from historic data which construct input-output mappings for the impending problem [14]. Recently, use of DL driven techniques to efficiently address key problems in wireless networks have generated interest in the research community. For example, the work in [10] has focused on DL based multicast traffic demand predictions to perform resource allocation for broadband networks. DL is used in [8] to predict average rates of non-realtime service users to assign radio resources in advance in a mobile network. In [15], a DNN has been used by the authors for subchannel and power allocation in non orthogonal multiple access (NOMA) networks. The authors in [16] have used a DNN for subcarrier assignment for users in an orthogonal frequency division multiple access (OFDMA) system. The work in [17] proposes a feed forward NN based resource and power allocation scheme for 4G LTE heterogeneous networks. The work in [18] proposes a DNN to learn optimal policy for predictive resource allocation with interference coordination for cellular networks. The authors in [19] present a DL based method to solve the problem of sub-band and power allocation in a multi-cell network. All of the preceding works are based on the assumption of similar distribution for the training and target datasets which is not always true in reality [20]. Therefore, those works can exhibit suboptimal performance when deployed in a real wireless environment [21].

Transfer learning (TL) is a concept in ML which can be used to make more effective predictors in a domain with limited training data availability by training the model beforehand in a domain where it is readily available [22]. TL can be used to address the problem of data distribution change over time. Nevertheless, TL has been typically performed

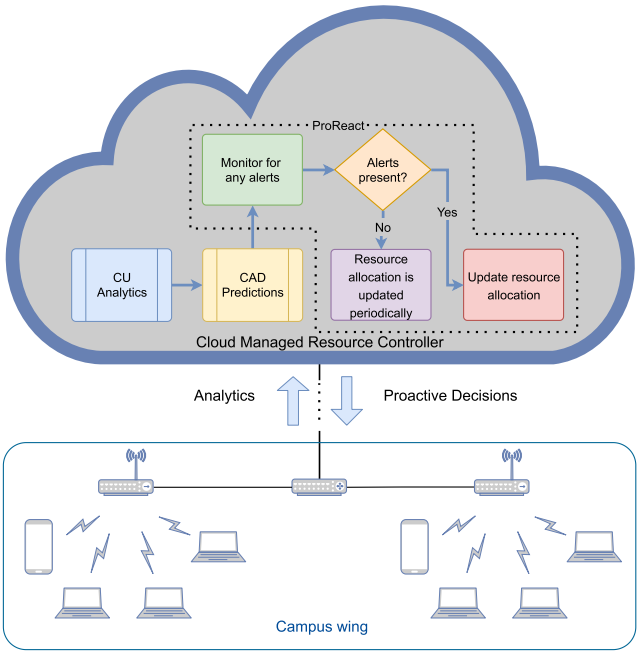


FIGURE 2. Proposed proactive resource allocation driven network architecture.

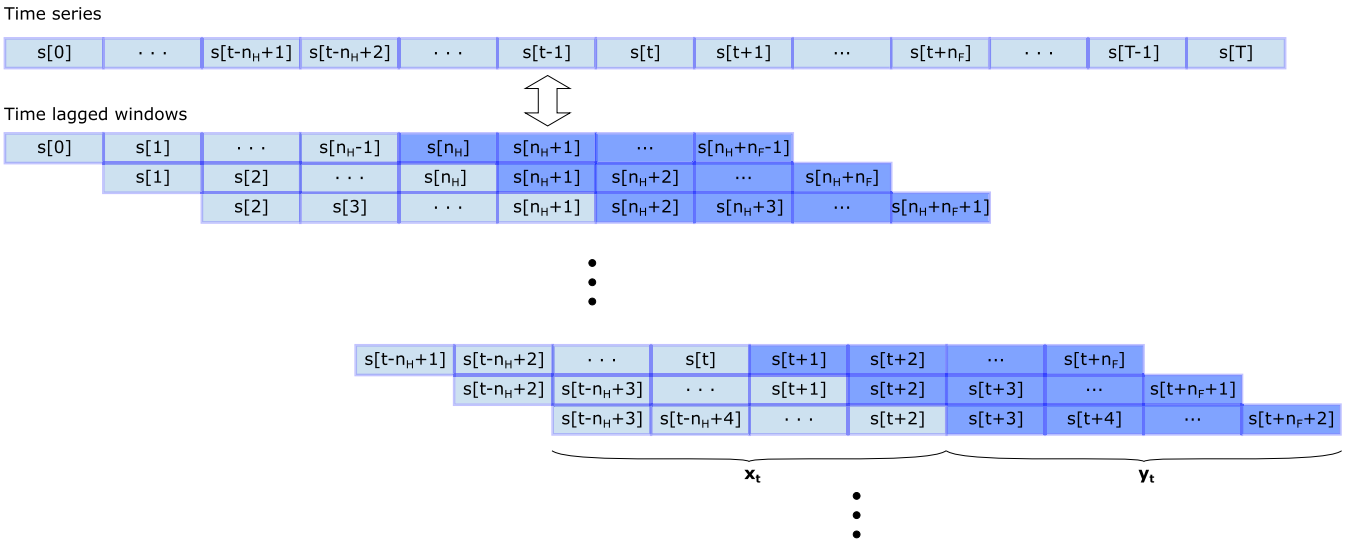


FIGURE 3. Feature extension using sliding window approach.

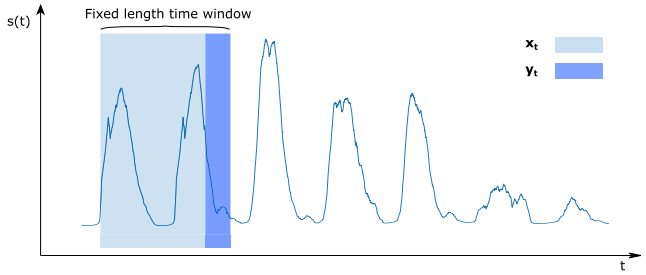


FIGURE 4. A window of fixed size is moved across time axis to generate $\mathbf{x_t}$ and $\mathbf{y_t}$ vectors.

offline which has limited its usage in online and real-time applications [23]. It is important to note that unlike our work, most of the DL based wireless resource allocation solutions in the literature leverage only synthetic datasets rather than using both synthetic and real world wireless network dataset. Furthermore, most of the works use single point predictions on the time series of considered key data metric without capturing any uncertainty in the predictions. The work in [24] presents a theoretical framework based on Bayesian inference for determining model uncertainty in NNs using dropout. The work in [11] further explores other kinds of uncertainties associated with NNs and proposes a framework to capture them systematically. Capturing uncertainties via the use of a PI is better in the sense that it gives extra freedom to determine up to which level the predictions from the model can be trusted. Besides, our work shows that uncertainty aware predictions can be used to improve a wireless resource allocation algorithm as the uncertainty estimate from the prediction model can be used by the algorithm to deploy alerts for possible reallocation of channel resources when resource utilization at an AP exhibits unusual behavior.

Recently, cloud managed networks have established themselves as efficient players in the operation and management of medium to large-scale deployment of wireless networks [25], [26]. A cloud managed network can collect descriptive analytics to track the right metrics and use DL on metrics data to make predictions that can be used to improve resource allocation in such networks. Our proposed DL driven resource allocation algorithm for cloud managed enterprise networks in unlicensed shared spectrum bands, such as a network deployed in a university campus, a large office building, or an airport etc. The proposed algorithm not only performs proactive resource allocation for multiple APs based on predictions from the developed BNN, but also keep tracks of uncertainty due to model misspecification in real-time. The uncertainty estimate from the prediction model is used to improve the performance of the resource allocation algorithm. To the best of our knowledge, this is the first time uncertainty aware predictions have been applied in a wireless resource allocation scenario.

III. SYSTEM MODEL

We consider a set of n APs denoted by $\mathcal{N} = \{1, 2, \cdots, N\}$ and a set of M unlicensed channels denoted by $\mathcal{M} = \{1, 2, \cdots, M\}$. The M channels are utilized by the APs of the enterprise wireless local area network (WLAN). Each AP in the network is denoted by α_i which represents the i^{th} AP. The ne'work's resource allocation is managed by a cloud managed resource controller. The basic system model illustrating a wireless network with predictions/resource allocation modules is presented in Fig. 2.

Enterprise WLANs often exhibit patterns over certain time periods, such as over a length of a day, a week, etc. in terms of CU. However, although the CU often has recurring patterns, they can be affected by uncertain events, such as abrupt increase in channel usage by wireless users within a short period of time. Our goal is to use a confidence aware deep (CAD) predictions technique that can not only predict wireless CU values but can also reliably estimate uncertainty in predicted values. The uncertainty estimates allow us to quantify how much to trust the predictions produced by the

TABLE 1. List of some of the important symbols used in the paper.

Meaning	Symbol
Number of APs	N
Set of APs	$\mathcal{N}$
Number of channels	M
Set of channels	$\mathcal{M}$
i^{th} AP	α_i
Robustness measure	$\mathcal{R}$
DL model	$\mathcal{D}$
Individual metric score of α_i on channel k	I_i^k
Marginal metric score of α_i on channel k	MS_i^k

DL model. Our goal is also to present an application of the CAD predictions to a proactive channel allocation algorithm.

A. REAL ENTERPRISE WLAN DATA

In our work, along with synthetic data we also use real wireless CU data. To consider a real enterprise WLAN, we collected CU data over a period of 5 weeks in the busiest parts of the University of Oulu. CU is an important wireless physical layer resource utilization metric to get information about the health of an enterprise WLAN [6], [27]. We measured the CU using hardware-accelerated spectrum analytics device implemented by us on a X'linx's Zynq-7000 system on chip (SoC) devices [13]. Each implemented device outputs every 20 seconds a measured CU value for that time duration. Hence, our datasets are CU time series.

B. PREDICTIONS/RESOURCE CONTROLLER

The cloud managed resource controller system shown in Fig. 2 collects the time series CU data from the CU analytics devices periodically, every 20 seconds. The CAD predictions model generates future CU predictions and their uncertainty estimates, for a specific time interval, and delivers them to the channel allocation module called *ProReact*. The module utilizes predictions and their uncertainty estimates to not only perform proactive allocations for multiple APs periodically, but also to adjust allocations to any significant changes in wireless CU instantly.

IV. ENCODER DECODER BASED DEEP LEARNING MODEL

We use an encoder-decoder based recurrent BNN to build the DL model as it is a type of NN well-suited to predict not only time series values, but also uncertainty estimates with the prediction values. The encoder-decoder network processes a time series step-by-step, maintaining an internal state summarizing the information it has seen so far. Over a period of time, it tries to learn what to keep and how much to keep from the past, and how much information to keep from the present state, which makes it so powerful as compared to the other NNs.

A. MODEL INPUT

Consider a univariate CU time series $\mathbf{s} = \{s[0], s[1], s[2], \cdots, s[T-1]\}$. To formulate the prediction problem as a supervised machine learning problem, we adopt a sliding window approach. A regressor vector $\mathbf{x_t}$ is composed by sliding a fixed window of size n_H across the time series which generates sequences of time lagged data as shown in Fig. 3 and 4. The generated sequence is given as input to a predictor f^θ which is parameterized on θ which aims to forecast the next n_F values of the time series.

The regressor vector at discrete time t is defined as $\mathbf{x_t} = \big[s[t-n_H+1], \cdots, s[t]\big] \in \mathbb{R}^{n_H}$. The predictor f^θ needs to infer the next n_F samples represented by the vector $\mathbf{y_t} = \big[s[t+1], \cdots, s[t+n_F]\big] \in \mathbb{R}^{n_F}$. Similarly, we can denote the inference from the predictor as $\hat{\mathbf{y}}_t = f^\theta(\mathbf{x_t}) \in \mathbb{R}^{n_F}$.' Let's assume the exact function which maps input vector $\mathbf{x_t}$ to the output vector $\mathbf{y_t}$ is $\bar{f}$, then, $\forall t,\ \mathbf{y_t} = \bar{f}(\mathbf{x_t})$. Then, when training the model, the learning algorithm would adapt the parameter θ to approximate f^θ to $\bar{f}$ as far as possible based on some performance metric. By using *mean average error* (MAE) as the performance metric in training the DL model, the MAE loss function for the learning problem is given by

$$J(\theta) = \frac{1}{|\mathbb{X}|} \sum_{\mathbf{x_t} \in \mathbb{X}} |\bar{f}(\mathbf{x_t}) - f^\theta(\mathbf{x_t})|, \tag{1}$$

where $\mathbb{X}$ denotes the set of regressor vectors. Supervised learning always solves an optimization problem to find the optimal function which minimizes the loss function given by (1). Let the optimal function be $f^{\hat{\theta}}$, then using (1), the optimization problem for the training can be formulated as

$$f^{\hat{\theta}} = \arg\min_{\theta}\ J(\theta). \tag{2}$$

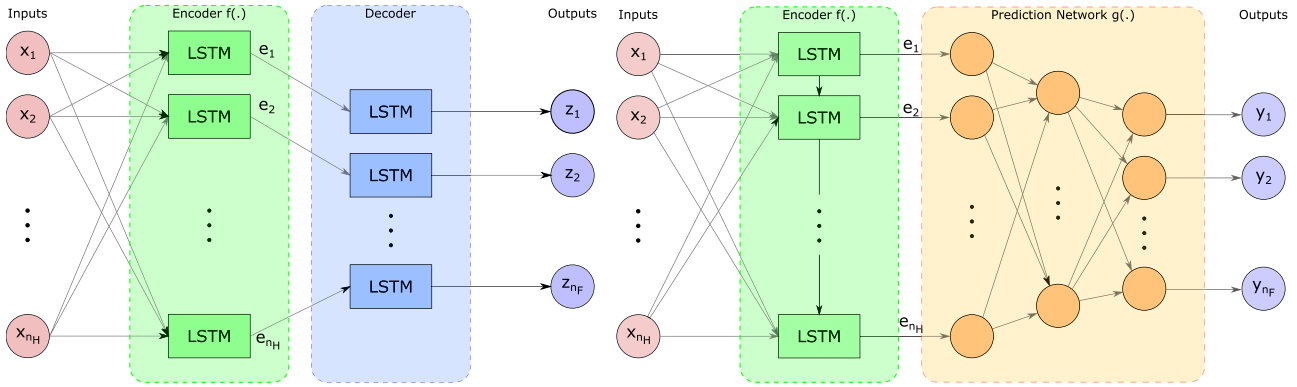
B. MODEL UNCERTAINTY AND INHERENT NOISE

Let $f^{\hat{\theta}}$ be the trained DL model with $\hat{\theta}$ representing the fitted weights. For a new sample point x^*, the prediction from the model is given by $y^* = f^{\hat{\theta}}(x^*)$. By computing the standard error σ of the predictions, the uncertainties associated with the model can be captured. Then, the resulting PI can be constructed as $[y^* - z_{\alpha/2}\sigma,\ y^* + z_{\alpha/2}\sigma]$ where $z_{\alpha/2}$ corresponds to the upper $\alpha/2$ quantile of the standard normal distribution.

The Bayesian probability theory provides a robust approach to address and quantify the uncertainties associated with a DL model. Let X, Y denote the observations used to train the model. Using Bayesian probability theory, the predictive probability density of the model for a new data point x^* can be obtained by,

$$p(y^*|x^*) = \int_\theta p(y^*|f^\theta(x^*)) \times p(\theta|X, Y)\, d\theta. \tag{3}$$

Estimating the posterior density $p(\theta|X, Y)$ is important in accurately quantifying the prediction uncertainty. Various inference methods are available to approximate the posterior density in DL models. Due to its simplicity in implementation, we use Monte Carlo (MC) dropout to approximate the model uncertainty. Dropout is the process of randomly dropping out hidden units in a DL model with certain probability p. By applying dropout stochastically for K times at



(a) Encoder-decoder network which is used to extract representative features from the time series (b) Inference network created by combining the encoder and prediction network

FIGURE 5. Model in (a) is trained first. After training, the encoder part is plugged to the prediction network to create the final CAD prediction model as shown in (b).

testing, the uncertainty associated with the predictions can be quantified as follows.

$$\begin{aligned} Var(y^*|x^*) &= Var[\mathbb{E}(y^*|\theta, x^*)] + \mathbb{E}[Var(y^*|\theta, x^*)], \\ &= Var\big(f^{\theta}(x^*)\big) + \sigma^2, \\ &\approx \frac{1}{K}\sum_{k=1}^{K}(\hat{y}^*_{(k)} - \bar{\hat{y}}^*)^2 + \sigma^2, \end{aligned} \tag{4}$$

where $\hat{y}^*_{(k)}$ is the model output at the k^{th} stochastic run with dropout applied and $\bar{\hat{y}}^*$ denote the mean of K outputs. The variance in (4) consists of two terms which correspond to *model uncertainty* and *inherent noise* respectively. The inherent noise term, σ^2 can be estimated by an independent validation set. Let the validation set b' $X' = \{x'_1, \cdots, x'_V\}$, $Y' = \{y'_1, \cdots, y'_V\}$ and the trained model on the training data be $f^{\hat{\theta}}(.)$, then σ^2 is estimated by

$$\sigma^2 = \frac{1}{V}\sum_{v=1}^{V}\big(y'_v - f^{\hat{\theta}}(x'_v)\big)^2. \tag{5}$$

C. ENCODER-DECODER FRAMEWORK FOR MODEL MISSPECIFICATION

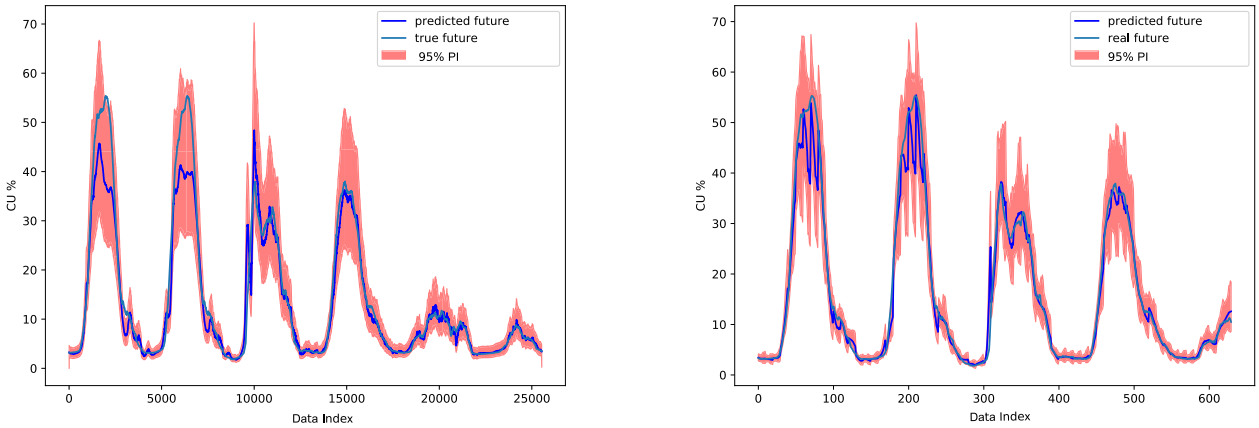
We capture the model misspecification by using an encoder-decoder framework formed using LSTM layers. When the encoder-decoder framework is trained on the training data, a latent embedding space is created by the encoder which extracts different features from the timeseries. If the test data have patterns different from training data, the encoder would not be able to correctly map them to the latent embedding space. Therefore, by pre-training the encoder-decoder framework, we can quantify the uncertainty due to model misspecification. Fig. 5(a) shows the encoder-decoder network used at pre-training phase.

The uncertainty in variance calculation is assimilated by connecting the encoder with a prediction network and treating the resulting network as a single network which we call as inference network. Let $f(.)$ be the encoder model and $g(.)$ be the prediction network, then the resulting inference network $h(.)$ can be written as the composite model $h(.) = g\big(f(.)\big)$. Fig. 5(b) shows the inference network created in this way. Let the input sequence vector to the model be $\mathbf{x} = (x_1, \cdots, x_{n_H})$, then the encoder forms the vector $\mathbf{e} = f(\mathbf{x})$ in the latent embedding space and the prediction network g generates the final output taking the vector $\mathbf{e}$ as the input to the network. In each forward pass, MC dropout is applied stochastically to all layers both in the encoder and the prediction network for K times. Applying dropout randomly in the encoder captures the uncertainty due to model misspecification. The dropout applied in the LSTM layer in the encoder is for both the input and the recurrent states.

D. MODEL IMPLEMENTATION FOR REAL CU DATASET

The encoder-decoder network is formed for the real CU data using two LSTM layers which consists of 32 LSTM cells in the first layer (which gives a dimension of 32 for the latent embedding space) and 10 LSTM cells in the second layer with *tanh* activation in all layers. The prediction network consists of three fully connected layers with 32, 16 and 10 hidden units with *tanh* activation in each layer respectively. The number of layers, LSTM cells and hidden units are selected heuristically to obtain the best prediction performance. We use a sliding window as shown in Fig. 3 to generate $\mathbf{x_t}$ and $\mathbf{y_t}$ vectors and use them to train the network. The steps involved in the model implementation are presented in Algorithm 1.

The training of the network takes place in two phases. In the first (pre-training) phase, the encoder-decoder network is fitted to the training data. Let the input sequence vector of the univariate time series at time t be $\mathbf{x_t} = \big[s[t - n_H + 1], \cdots, s[t]\big]$, the encoder takes $\mathbf{x_t}$ and maps that to a low dimensional vector $\mathbf{e_t}$ in the embedding space. Decoder then learns to recreate the output sequence vector $\mathbf{y_t} = \big[s[t+1], \cdots, s[t+n_F]\big]$ from $\mathbf{e_t}$. This way, the encoder learns to extract relevant features present in the input time series. In the second phase, we use the encoder to encode the input vector $\mathbf{x_t}$ to $\mathbf{e_t}$ and we train the prediction network



(a) 95% PI for single time-point prediction

(b) 95% PI for multiple time-point prediction

FIGURE 6. **Prediction values and 95% PIs for single and multiple time-point predictions using the proposed CAD predictions model.**

to predict $\mathbf{y_t}$ using $\mathbf{e_t}$ as the input. Once the training of the prediction network is finished, we cascade the encoder with the prediction network and form the inference network as shown in Fig. 5(b). Note that for the same inputs, the outputs from the encoder-decoder network and the inference network would be different as shown in Fig. 5. We use MC dropout as discussed in section IV-B to quantify the uncertainty and generate the PIs. Selection of values for K and p is heuristic. K should be selected in such a way that it generates a smooth PI. Nevertheless, having a very large value for K greatly increases computational time. In our case, we used $K = 100$. For p, $0 < p \leq 0.5$ is held. The value of p can be selected heuristically by calculating the resulting empirical coverage. In statistics, empirical coverage of predictions is defined as the proportion of the samples of interest which would be contained in the PI. p is selected such that the correct empirical coverage is obtained.

We set the sliding window size to cover 7 days of CU data and use subsampling on the data with a factor of 32. The models are trained to predict a time horizon of 1.5 hour. We train all the models using *RMSprop* optimizer with MAE selected as the loss function. All the models are implemented using *TensorFlow 2.0.0* machine learning platform with *Python 3.7* environment.

E. PREDICTION MODEL RESULTS

In this section, we use real CU data to evaluate the performance of the CAD model with respect to prediction and uncertainty. Prediction performance is evaluated by comparing the results with several other prediction models.

1) UNCERTAINTY ESTIMATION

We measure the performance of the model in estimating the uncertainty in the predictions by calculating the empirical coverage. We use the MC dropout probability, p and calculate the resulting empirical coverage of the calculated PI. For $p = 0.5$, the resulted empirical coverage for different

TABLE 2. **Empirical coverage for the proposed model.**

Standard score	Expected PI	Empirical coverage
1.96	95%	94.53%
1.64	90%	91.24%

standard score values is given in Table 2. Standard score is the number of standard deviations a sample value is above or below the mean value. For a sample value x, it is calculated as $(x-\mu)/\sigma$ where μ is the mean and σ is the standard deviation. In the table, we can see that the empirical coverage values calculated for PIs from our proposed model are very closer to the expected PIs. Next, we define single and multiple time-point predictions.

Definition 1: Prediction made by a predictive model for a specific time point in the future is defined as a ***single time-point prediction****.*

Definition 2: Predictions made by a predictive model for multiple time points in the future are defined as ***multiple time-point predictions****.*

Fig. 6(a) and Fig. 6(b) show the real CU values, predicted CU values and calculated PIs using MC dropouts for the single time-point predictions (the point at 1.5 hour in the future) and multiple time-point predictions for the test data set. In single time-point predictions, the CAD model makes predictions every sample period where as in multiple time-point predictions, the CAD model makes predictions every 1.5 hours. In Fig. 6(b), we can see that in multiple time-point predictions, prediction values can fluctuate compared to single time-point predictions. In Fig. 6(b), it is apparent that the PI widens when going from the closest to the furthest point in a multiple time-point prediction. Moreover, we can observe in the figures that most of the time, the actual value falls inside the 95% PI calculated by our model. Also, in Fig. 6(a), we can see that our proposed model gives a broad PI at the peaks of the CU time series. It makes sense because at peaks, the

Algorithm 1 CAD Prediction Model Implementation

Input: CU time series denoted by $\mathbf{s}$

// Prepare the dataset
$\mathbf{s}' \leftarrow Filter(\mathbf{s})$ // Moving average filtering of CU time series
$\mathbf{x}, \mathbf{y} \leftarrow Databatch(\mathbf{s}')$ // Data batching with decimation
$\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, x', y', x^*, y^* \leftarrow Partition(\mathbf{x}, \mathbf{y})$ // $(\tilde{x}, \tilde{y})$ = train split, (x', y') = validation split, (x^*, y^*) = test split

// Generate Encoder-Decoder network and train
$Encoder(\cdot) \leftarrow LSTM(\cdot)$ // *LSTM* layer with *tanh* activation
$Decoder(\cdot) \leftarrow LSTM(\cdot)$
$AutoEncoder(\cdot) \leftarrow Decoder(Encoder(\cdot))$
$Train(AutoEncoder, \tilde{\mathbf{x}}, \tilde{\mathbf{y}})$ // *RMSprop* optimizer

// Generate prediction network and train
$PredictionNetwork(\cdot) \leftarrow Dense(\cdot)$ // *Dense* layer with *tanh* activation
$\tilde{\mathbf{e}} \leftarrow Encoder(\tilde{\mathbf{x}})$
$Train(PredictionNetwork, \tilde{\mathbf{e}}, \tilde{\mathbf{y}})$ // *RMSprop* optimizer

// Generate Inference network
$InferenceNetwork(\cdot) \leftarrow PredictionNetwork(Encoder(\cdot))$

// Calculate ${\sigma_1}^2$
$\hat{\mathbf{y}}' \leftarrow InferenceNetwork(\mathbf{x}')$
${\sigma_1}^2 = \frac{1}{|\mathbf{y}'|} \sum_{v \in |\mathbf{y}'|} \left(\hat{y_v}' - y_v'\right)^2$

// Generate dropout network with dropout probability, p
$MCDropoutNetwork \leftarrow Dropout(InferenceNetwork, p)$

// Calculate ${\sigma_2}^2$ using MC dropout
$\hat{\mathbf{y}}^* = InitializeToEmptyArray()$
for $k = 1$ to K **do**
  $\hat{\mathbf{y}} \leftarrow MCDropoutNetwork(\mathbf{x}^*)$
  $\hat{\mathbf{y}}^* \leftarrow [\hat{\mathbf{y}}^*, \hat{\mathbf{y}}]$
end for
$\sigma_2^2 = \frac{1}{K} \sum_{k=1}^{K} \left(\hat{\mathbf{y}^*}_k - \bar{\hat{\mathbf{y}}}^*\right)^2$ where $\bar{\hat{\mathbf{y}}}^* = Mean(\hat{\mathbf{y}}^*)$
$\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$

// Calculate prediction and PIs
$\hat{\mathbf{y}} \leftarrow InferenceNetwork(\mathbf{x}^*)$
$\hat{\mathbf{y}}_{pi\_lb} = \hat{\mathbf{y}} - z_{\alpha/2}\sigma$ // lower boundary
$\hat{\mathbf{y}}_{pi\_ub} = \hat{\mathbf{y}} + z_{\alpha/2}\sigma$ // upper boundary

Output: $\hat{\mathbf{y}}$, $\hat{\mathbf{y}}_{pi\_lb}$, $\hat{\mathbf{y}}_{pi\_ub}$

prediction uncertainty is high which can be explained by the phenomena of model uncertainty and model misspecification.

2) PREDICTION PERFORMANCE OF THE PROPOSED MODEL

To evaluate the prediction performance, we compare the results to three different models; i) a dInaive predictor: the forecasts for a given day are equal to the values of a full day before, e.g. predictions for Tuesday, Feb. 19 are equal to the values of time series on Monday, Feb. 18, ii) a vanilla LSTM model which uses 32 LSTM units in the first layer preceded by fully connected dense layers with 32, 16 and 10 units, and iii) a GRU model which has the same structure as the vanilla LSTM model. Fig. 7 shows the training losses of the

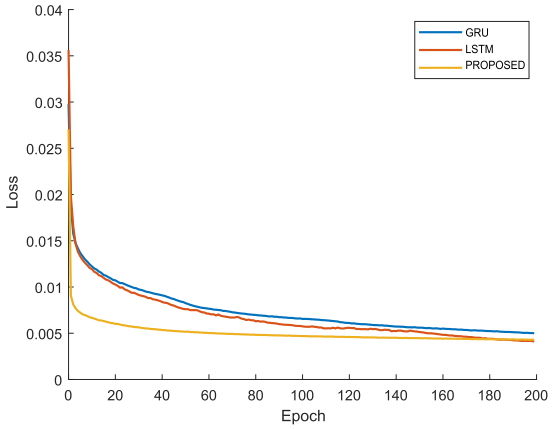


FIGURE 7. Training losses of the different models.

TABLE 3. MAPE of different models using multiple time-point predictions for original test data and test data included with unseen samples.

Model	MAPE (original test data)	MAPE (test data with unseen samples)	Robustness measure, $\mathcal{R}$
Daily naive	26.81%	45.50%	0.5892
LSTM	7.28%	11.73%	0.6206
GRU	6.69%	13.19%	0.5072
Proposed	8.02%	9.84%	0.8150

LSTM model, GRU model and the proposed model. From the figure, we can identify that the proposed model converges faster with lesser number of epochs than other two models and the training loss is lower than the GRU model and almost similar to the LSTM model after the lapse of 180 epochs.

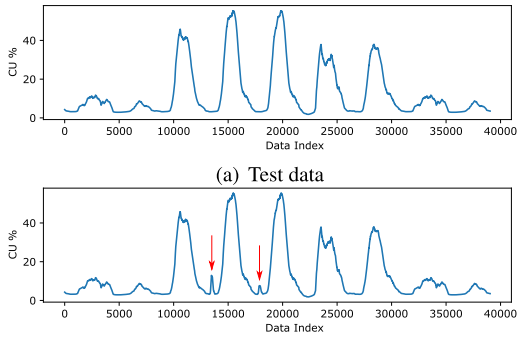
To compare the prediction accuracy of the models, we use the measure called MAPE which is given by

$$MAPE = \frac{100\%}{N} \sum_{i=1}^{N} \left| \frac{y_i - \hat{y}_i}{y_i} \right|, \tag{6}$$

where y_i is the actual value and $\hat{y}_i$ is the predicted value. Each model makes multiple time-point predictions with a forecast horizon of 1.5 hours and MAPE score for the predictions are calculated using (6), where forecast horizon is defined as,

Definition 3: Future period of time for which forecasts are generated is defined as ***forecast horizon****.*

To quantify the model robustness, we propose a metric called robustness measure. Robustness measure quantifies the model performance in proportion to its performance under a disturbance, such as presence of unseen new samples in test data. Let us denote a DL model by $\mathcal{D}$, model input by X, observed values by Y and the model output by $\hat{Y}$. The output of the model and observed values under a disturbance δ is denoted by $\hat{Y}_\delta$ and Y_δ, respectively. Then, the robustness measure $\mathcal{R}$ is defined by $\mathcal{R}(\mathcal{D}|X, Y, \delta) = \frac{\rho(\hat{Y}, Y)}{\rho(\hat{Y}_\delta, Y_\delta)}$ where ρ can be a performance measure such as MAPE. A model



(a) Test data

(b) Test data with new unseen samples (indicated by arrows)

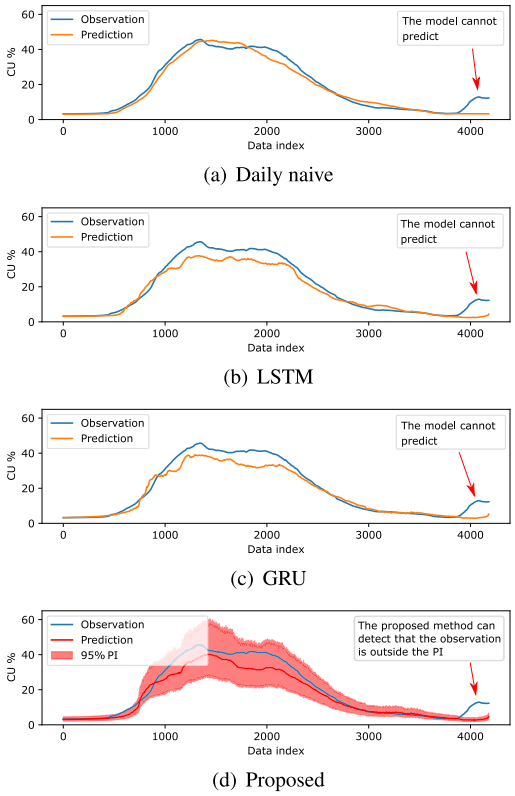
FIGURE 8. (a) shows the original test data. (b) shows the test data included with new unseen samples.

having $\mathcal{R}$ close to 1 is more robust as it performs better even under disturbances.

To compare the model performance, we use test data as shown in Fig. 8(a) and new test data which includes new unseen samples with a different distribution than the data distribution available at DL model training time as shown in Fig. 8(b). The duration of the introduced new test data is around two hours. According to Table 3, it is seen that the MAPE of the proposed model for original test data is comparable to LSTM and GRU models. For new test data with unseen samples, we can see that the proposed model outperforms other models in terms of MAPE. Furthermore, we see that the performance degradation of the proposed model in the presence of unseen samples is the lowest which results in the best robustness measure across all the tested models. This concludes that the proposed model has the most stable and robust predictive performance compared to other benchmark models.

Fig. 9(a), 9(b), 9(c) and 9(d) show how each model behaves in the presence of new unseen samples which represent change in CU distribution. Although the proposed DL model showed improvement in terms of MAPE, it is clear from the figures that no model can correctly predict the future in the presence of test data which represent the change in distribution. It makes sense due to the fact that new test samples were not present in the training data. This shows that a change in data distribution significantly affects the performance of DL models.

Further, it can be observed in Fig. 9(d) that the observed values lie outside the PI generated by our proposed model. This helps us to identify the change in the new test data as a change in the CU distribution. In the next section, we use this feature to trigger alarms for the improvement in channel reallocation in a real enterprise WLAN.



(a) Daily naive

(b) LSTM

(c) GRU

(d) Proposed

FIGURE 9. (a), (b), (c) and (d) show the predictions from each model in the presence of new unseen test data.

V. APPLICATION OF CAD PREDICTIONS TO WIRELESS RESOURCE ALLOCATION

In this section, we present an application of the developed CAD predictions model in frequency channel resource allocation for a cloud managed WLAN operating in an unlicensed spectrum. The proposed resource allocation framework jointly addresses two QoS criteria: 1) channel quality by taking into account signal-to-interference-noise ratio (SINR); and 2) the amount of airtime required by an AP for a variety of wireless applications by taking into account its CU demand. Based on the individual and group preferences, we design two metric scores that take into account the CU predictions from the CAD model. The metric scores are utilized to perform channel allocation decision in the proposed algorithm called *ProReact*. The steps involved in the algorithm are presented in Algorithm 2. The main concept behind the proactive channel allocation of the proposed algorithm can be summarized as follows: The cloud controller periodically collects the CU data from the APs in each channel k. The CAD predictions model calculates CU PI upper bounds denoted by $\hat{\mathbf{y}}^k_{pi\_ub}$ for the next allocation period. The controller then utilizes a metric score which takes into account the maximum of the obtained $\hat{\mathbf{y}}^k_{pi\_ub}$ and the data transfer rates at APs denoted by R^k_i to generate a new proactive channel plan $\mathcal{S}$ for the next allocation period and delivers the updated configuration to the APs. This proactive channel allocation process is denoted by line numbers 4-6 in Algorithm 2.

In wireless networks, CU may exhibit different behavior than usual due to the dynamicity in the usage demand of the users connected to the AP. Due to this reason, a proactive channel assignment algorithm based only on predicted CU values (we use the maximum of $\hat{\mathbf{y}}^k_{pi\_ub}$, see line 4-5 of

Algorithm 2 *ProReact* Channel Allocation Based on Individual and Marginal Metric Scores Based Decisions

1: **Input:** p (Dropout probability) , $\mathbf{x_t}^k$ (CU time series regressor vector for channel k), A^k (Total CU demand of APs in channel k), $\hat{A}_i^k$ (CU demand of α_i in channel k), R_i^k (rate of α_i in channel I) $\forall i \in \mathcal{N},\ \forall k \in \mathcal{M}$
// Initial channel allocation based on current CU demands
2: $\mathcal{S} :=$ AllocateChannels$(A^k, \hat{A}_i^k, R_i^k)$

// CAD predictions based channel allocation
3: **for** each Allocation Period **do**
// Obtain the PI upper bound using MC dropouts
4: $\hat{\mathbf{y}}_{pi\_ub}^k \leftarrow Dropout(InferenceNetwork(\mathbf{x_t}^k), p)$
5: $\breve{A}^k = max(\hat{\mathbf{y}}_{pi\_ub}^k)$;
// Proactive channel allocation based on the maximum of
// predicted CU PI upper bound
6: $\mathcal{S} :=$ AllocateChannels$(\breve{A}^k, \hat{A}_i^k, R_i^k)$
7: **while** *True* **do**
// Monitor for any alerts
8: **if** $A^k > \breve{A}^k$ **then**
// Reactive channel allocation based on real-time CU
9: $\mathcal{S} :=$ AllocateChannels$(A^k, \hat{A}_i^k, R_i^k)$
10: **else**
11: **Go To** line 3 at the end of current allocation period
12: **end if**
13: **end while**
14: **end for**

15: **function** AllocateChannels$(A^k, \hat{A}_i^k, R_i^k)$
16: $AP\_list \leftarrow$ Random AP order
17: **for** each AP, $i \in AP\_list$ **do**
18: **for** each channel, $k \in \mathcal{M}$ **do**
19: Calculate A_i^k, See (8)
// Compute metric score
20: **if** INDIVIDUAL_METRIC_SCORE **then**
21: $I_i^k = \frac{A_i^k}{\hat{A}_i^k} \times R_i^k$, See (7)
22: **else**
23: $\Delta MS_i^k = MS_i^{\tilde{k}} - MS_i^k$, See (9)
24: **end if**
25: **end for**
26: Perform best response update
27: Update channel plan $\mathcal{S}$
28: **end for**
29: **return** S
30: **end function**

Algorithm 2) may not be sufficient to ensure a good real-world performance. To overcome this limitation, our proposed algorithm incorporates the maximum of PI estimate denoted by $\breve{A}^k$ to raise an alert for possible channel reallocation in real-time. In each allocation period, the cloud controller observes whether the real-time CU denoted by A^k in a channel exceeds $\breve{A}^k$. If the real-time CU is outside $\breve{A}^k$, the cloud controller allocates channels reactively by generating a new channel plan $\mathcal{S}$ using the real-time CU and delivering the updated configuration to the APs. This reactive channel allocation process is denoted by the line numbers 8 and 9 in Algorithm 2.

Basically, there are two major advantages in utilizing the PIs provided by BNN in channel allocation; PIs allow the channel allocation algorithm, 1) to take into account various kinds of uncertainties associated with the predictions when performing channel allocation and, 2) to adapt its channel allocation decisions to anomalous CU levels.

A. PROPOSED METRIC SCORES

We present the following individual metric which is utilized by the cloud controller to evaluate the effectiveness of an AP when allocated to a channel k. The individual metric score is meaningful from the perspective of the cloud performing channel selections in a wireless system as it captures usefulness in terms of CU and rate an AP gets from the allocated channel action. Higher values of the metric imply that the AP will have a better wireless experience. The presented metric I_i^k estimates the effectiveness of α_i on a channel k as

$$I_i^k = \frac{A_i^k}{\hat{A}_i^k} \times \left[B_i\ log(1 + SINR_i)\right], \tag{7}$$

where B_i and $SINR_i$ denote the bandwidth and the SINR of α_i on channel k respectively, $\hat{A}_i^k$ denotes the CU demand of α_i on channel k (which is generated by the applications running on the devices connected to α_i), and A_i^k is the CU obtained by α_i allocated to channel k. A_i^k can be calculated as [28],

$$A_i^k = \begin{cases} \hat{A}_i^k, & \text{if } \sum_{j \in \mathcal{C}^k} \hat{A}_j^k \leq A^k \\ min\left\{\hat{A}_i^k, \frac{1}{|\mathcal{C}^k|}\right\}, & \text{otherwise} \end{cases} \tag{8}$$

where A^k is the total obtainable CU on channel k and $\mathcal{C}^k$ represents the set of APs which are present in channel k. A_i^k can be explained as follows. When the sum of total CU demands of APs in a channel k is less than or equal to the total available CU, then the obtained CU of α_i is equal to its CU demand. However, in a channel, when the total CU demand of APs exceeds A^k, then α_i can still expect to get its fair share of the CU which is at most $1/|\mathcal{C}^k|$.

The individual metric score I has two important properties: i) If an AP experiences high interference on channel k, then its rate will decrease and hence the score I will decrease; ii) I can only increase in terms of obtained CU as long as it is less than the CU demand. When the CU demand is satisfied, then I cannot further increase in terms of obtained CU.

Let the current channel of α_i be k and any other channel which is available for the cloud controller to select for α_i be $\tilde{k}$, then the individual metric scores of α_i on each channel can be given as I_i^k and $I_i^{\tilde{k}}$ respectively.

The second metric score we consider is called the marginal metric score which is denoted by MS_i^k. The marginal metric score has an important property that it takes into account the sum of individual metric scores of all APs present in the channel. The marginal metric score of α_i with respect to channel k is defined as the difference between the sum of the individual metric scores of APs with α_i present in the channel and without α_i in the channel. The marginal metric score of

α_i can be given as

$$MS_i^k = \sum_{j \in \mathcal{C}^k \cup i} I_j^k - \sum_{j \in \mathcal{C}^k} I_j^k. \quad (9)$$

B. ALLOCATION DECISION RULES AND BEST RESPONSE

Before presenting the allocation decision rules, we first present some definition related to the concept of best response (BR) updates.

Definition 4: Let us consider an allocation update which involves cloud controller changing the channel of α_i, whi'e all other APs' allocations are kept unchanged. Then we say that an allocation for α_i is a better response update when its metric score is strictly increased due to the change, i.e., $I_i^{\tilde{k}} > I_i^k$ or $MS_i^{\tilde{k}} > MS_i^k$. Moreover, an allocation update is a BR update if it improves 'he α_i's metric score to the maximum possible value among all better responses for α_i.

We consider and compare two different decision rules for the cloud controller to perform BR updates in the proposed algorithm 2: 1) Individual metric score based decision rule; and 2) Marginal metric score based decision rule.

1) INDIVIDUAL METRIC SCORE BASED DECISION

The BR update using this decision rule is calculated bas'd only on an AP's own metric score. Under this decision rule, the BR update for α_i, BR_i can be given as

$$BR_i = \{\tilde{k} \in \mathcal{M}\backslash k \mid I_i^{\tilde{k}} - I_i^k > 0 \; and \; I_i^{\tilde{k}} = max\{I_i^l - I_i^k; \quad \forall l \in \mathcal{M}\backslash k\}\}. \quad (10)$$

2) MARGINAL METRIC SCORE BASED DECISION

Under this decision rule, a BR update is performed by taking into account not only the individual metric score of α_i, but also the metric scores of all other APs which are affected by the BR update. Under this decision rule, the BR update for α_i can be given as

$$BR_i = \{\tilde{k} \in \mathcal{M}\backslash k \mid \Delta MS_i^{\tilde{k}} > 0 \; and \; \Delta MS_i^{\tilde{k}} = max\{MS_i^l - MS_i^k; \quad \forall l \in \mathcal{M}\backslash k\}\}. \quad (11)$$

Based on which type of decision rule is utilized in the cloud controller, the channel allocation plan is updated by performing the BR updates by the cloud controller. In section VI, we will evaluate the performance of the proposed algorithm through simulations.

C. CONVERGENCE ANALYSIS

In this section, we analyze the convergence of the proposed cloud-based channel allocation algorithm when using the marginal metric score based decision rule. A network of wireless APs can be seen at each time instant as an undirected graph in which the nodes represent wireless APs, and there is an edge between two nodes if the nodes are within the transmission range of each other. The resulting connectivity graph G is undirected because interference and airtime competition among two APs in an unlicensed channel in general form bidirectional link. We focus on a game theoretic concept of convergence called NE. The use of NE concept is suitable as it ensures that no AP has an incentive to unilaterally deviate from the given allocation. We next present some theorems and definitions related to game theory to support our claims.

Definition 5: A pure strategy NE is an action profile of players in a game in w'ich each player's action is a BR to the rest of t'e other players' actions. A formal definition of NE outcome corresponding to channel allocation can be given as follows: A channel allocation profile $\mathbf{a} = (a_1, a_2, \cdots, a_N)$ of an N AP cloud allocated solution is a NE if for each α_i, we have

$$MS_i(a_i, \mathbf{a}_{-i}) \geq MS_i(\acute{a}_i, \mathbf{a}_{-i}); \quad \forall \acute{a}_i \in A_I, \quad \forall i \in \mathcal{N}. \quad (12)$$

That is, the allocated channel a_i of each α_i is a BR to the allocations $\mathbf{a}_i$ of all other APs.

*Theorem 5.1: When the BR updates for the APs are performed by the cloud, then, the channel allocation under the decision rule based on marginal metric score is an **exact potential game**. Strictly speaking, the channel allocation algorithm based on marginal metric score results in an NE channel allocation profile.*

Proof: The work in [29] has shown that marginal contribution metric results in an exact potential game with the potential function W which holds the following mathematical relationship.

$$MS(\acute{a}_i, \mathbf{a}_{-i}) - MS(a_i, \mathbf{a}_{-i}) = W(\acute{a}_i, \mathbf{a}_i) - W(a_i, \mathbf{a}_{-i}); \quad \forall a_i, \acute{a}_i \in A_i \; and \quad \forall a_{-i} \in A_{-i}. \quad (13)$$

Moreover, there is *finite best-response improvement property* associated with every potential game which means if the cloud continues to perform the BR updates for the APs, then it would ultimately lead the system to a'pure NE. Each AP's improvement in a step is finite and such a sequence of steps by APs ends in an NE.

For the proposed channel allocation algorithm, its potential function W is given as

$$W = \sum_{j \in \mathcal{N}} I_j. \quad (14)$$

Using the BR updates, when the cloud selects a new channel $\tilde{k}$ for α_i currently in channel k, then the change in the potential function for this update only happens due to the allocation change in k and $\tilde{k}$. Therefore, the difference in potential function when the cloud selects a new channel $\tilde{k}$ can be given as

$$W_i^{\tilde{k}} - W_i^k = \sum_{j \in \mathcal{C}^{\tilde{k}} \cup i} I_j^{\tilde{k}} + \sum_{j \in \mathcal{C}^k} I_j^k - \Big[\sum_{j \in \mathcal{C}^k \cup i} I_j^k + \sum_{j \in \mathcal{C}^{\tilde{k}}} I_j^{\tilde{k}} \Big],$$
$$= \Big[\sum_{j \in \mathcal{C}^{\tilde{k}} \cup i} I_j^{\tilde{k}} - \sum_{j \in \mathcal{C}^{\tilde{k}}} I_j^{\tilde{k}} \Big] - \Big[\sum_{j \in \mathcal{C}^k \cup i} I_j^k - \sum_{j \in \mathcal{C}^k} I_j^k \Big],$$

$$= -MS_i^{\tilde{k}} - MS_i^k,$$
$$\implies \text{exact potential game.} \quad (15)$$

Hence, (15) shows that the allocation decisions based on the marginal metric score leads to an exact potential game. Moreover, based on finite BR improvement property for every potential game shown in [30], we can conclude that the cloud-based channel allocation algorithm using marginal metric score guarantees convergence to an NE.

□

Note that it is not the NN which is enabling the convergence, but the output of the NN given to the carefully designed resource allocation algorithm which is leading to the convergence. Not any NN based resource allocation solution will converge. We need to come up with a carefully designed metric score and a resource allocation algorithm with carefully designed decision steps. The proposed resource allocation algorithm utilizes the potential function property (refer to Theorem 5.1) and the best response dynamics (refer to (11)) to ensure that it will converge. An NN which can only give predictions cannot be used directly in the proposed method.

VI. PERFORMANCE ANALYSIS OF CAD PREDICTION BASED WIRELESS RESOURCE ALLOCATION ALGORITHM

For the simulations, we consider N APs to be allocated in M channels. We evaluate the performance of the proposed cloud-based channel allocation algorithm in terms of average sum metric and average sum rate of the APs. We simulate the proposed algorithm under two scenarios, with low CU demands and high CU demands in the network, where low CU demands take the values $\hat{A}_i^k \in (0, 0.6]$ and high CU demands take the values $\hat{A}_i^k \in (0, 0.7]$, respectively. Our results focus on showing convergence of the proposed method using individual and marginal metric scores given by (7) and (9) utilizing different initial channel allocation routines. We also show that the proposed channel allocation algorithm which optimizes the average sum metrics of the APs will also leads to the optimized average sum rates of the APs.

To assess the effectiveness of the proposed method in different scenarios relative to the optimal solution, we compare the convergence of the proposed method with optimal average sum metric obtained via the best NE. Note that it is shown in [30] that convergence to the best NE of a marginal contribution based solution leads to the optimal solution.

We also show the performance gain of the proposed CAD predictions based channel allocation method by using test data with different CU distribution in a WLAN.

A. CONVERGENCE RESULTS

We evaluate the performance of the proposed method under four scenarios; 1) initzero individual: where initially no AP is allocated to a channel and channel allocation is based on individual metric score; 2) initzero marginal: where initially no AP is allocated to a channel and channel allocation is based on marginal metric score; 3) initrandom individual: where initially APs are allocated to channels randomly with uniform distribution and channel allocation is based on individual metric score and 4) initrandom marginal: where initially APs are allocated to channels randomly with uniform distribution and channel allocation is based on marginal metric score.

Fig. 10(a)-(d) show the average sum metrics and average sum rates of the APs under the aforementioned scenarios. It can be seen from Fig. 10(a)-(d) that the better response updates lead to increases in both the average sum metric and the average sum rate as a function of steps until the equilibrium.

For both the low demand and the high demand cases shown in Fig. 10(a)-(d), we can see that the marginal metric score based method always converges closer to the optimal solution than the individual metric score based method. This can be explained as follows. In individual metric score based method, although an AP selects a channel which gives a better metric score for itself than the current channel, it might affect the performance of other APs in the selected channel which results in a lower sum metric whereas in marginal metric score based method, an AP only selects a channel if that results in a higher sum metric for all the APs in the selected channel. Accordingly, the channel allocation method based on the marginal metric score always outperforms the channel allocation method based on the individual metric score.

It can be seen from the Fig. 10 that irrespective of initial channel allocation method, the algorithm almost converges to the same solution for both metric scores. Increasing the number of APs and the number of channels result in increased average sum metrics and average sum rates in the system.

In Fig. 10, we can see that the average sum metric and the average sum rate plots have the same shape. This implies that optimizing the average sum metrics results in optimized average sum rates in APs.

Finally, from the results, we can conclude that the marginal metric score based method has the best overall performance.

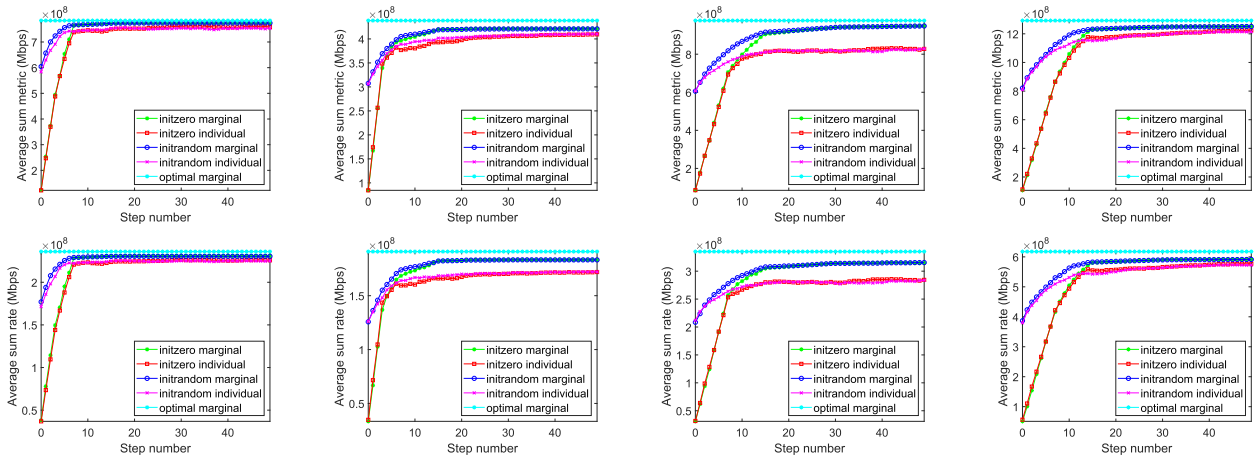
B. COMPLEXITY ANALYSIS

The complexity of the proposed Algorithm 2 in terms of single channel allocation step is within $\mathcal{O}(NM)$. Additionally, from Fig. 10, we can identify that the number of channel allocation steps taken by the algorithm to converge is less than $2N$ for all the considered scenarios. Therefore, the total complexity of the channel allocation algorithm is within $\mathcal{O}(N^2M)$.

C. THE PRICE OF STABILITY AND PRICE OF ANARCHY

As there can be more than one NE leading to convergence in the channel allocation solution, we need to evaluate the efficiency of the obtained NEs. There are two measures in game theory which correspond to the best and the worst achieved NE. The best and the worst NE are evaluated by the price of anarchy (POA) and the price of stability (POS), respectively. The POA and POS can be given as

$$POS = \frac{\text{value of best NE}}{\text{value of optimal solution}},$$



(a) $N = 8,\ M = 4$ Low Demand (b) $N = 8,\ M = 4$ High Demand (c) $N = 16,\ M = 8$ Low Demand (d) $N = 16,\ M = 8$ High Demand

FIGURE 10. (a)-(d) show average sum metrics and average sum rates of APs under different scenarios respectively.

TABLE 4. POA and POS results for various scenarios. *ind* and *marg* denote individual and marginal metric scores respectively.

Scenario	Score type	POA	POS
Case I: N=8, M=4 low demand	*ind*	0.9109	0.9970
	marg	0.9372	1
Case II: N=8, M=4 high demand	*ind*	0.8724	0.9964
	marg	0.9174	1
Case III: N=16, M=8 low demand	*ind*	0.8544	0.9743
	marg	0.8979	1
Case IV: N=16, M=8 high demand	*ind*	0.8526	0.9745
	marg	0.8963	1

$$POA = \frac{\text{value of worst NE}}{\text{value of optimal solution}}.$$

For the proposed channel allocation game, we have the following observation.

Observation 1: The proposed cloud-based channel allocation always achieves POS = 1 when using marginal metric score. This is in accordance to the result shown in [30] that convergence to the best NE of a marginal contribution based solution leads to the optimal solution.

As POA compares the worst NE to the optimal solution, it effectively evaluates the largest performance gap which is incurred by the channel allocation solution. This means that POA can be considered as a lower bound for the convergence performance of the channel allocation solution. Always the channel allocation solution would converge to an NE equal to or better than the NE corresponding to POA.

Table 4 lists the obtained POA and POS results for the various scenarios when utilizing marginal and individual metric scores for the BR updates. To calculate the results, we use simulations running over several MC runs and in each MC run, we repeat the channel allocation process for several steps. We calculate the results when $N = 8,\ M = 4$ and $N = 16,\ M = 8$ with APs experiencing low and high CU demands. In Table 4, we can see that in all the cases, the POA is greater than 0.85. This conveys the fact that even when the channel allocation converges to the worst NE, the degradation in performance (i.e. performance gap) compared to the global optimal solution is less than 15%. The other observation in Table 4 is that the POA when using marginal metric score is greater than the individual metric score. In Table 4, it can be seen that the calculated POS for individual metric score is very close to 1 which means that the best NE payoff when using individual metric score is close to the optimal solution. From the results in Table 4, we can conclude that utilizing the marginal metric score for BR updates gives the best performance in the channel allocation method.

D. PERFORMANCE EVALUATION USING CU DATA WITH NEW PATTERNS

We also evaluate the performance of the proposed CAD predictions based *ProReact* channel allocation method through simulations using real CU time series data. We consider a WLAN which consists of 8 APs and 3 unlicensed channels (e.g. $N = 8,\ M = 3$). To show the impact of CAD predictions on the resource allocation method, the real CU data set also contains a different CU pattern to the regular CU. The different CU pattern is used to model a change of CU distribution in the APs and its impact on the channel allocation. We test the algorithm by changing the duration of the different CU pattern under two cases (as shown in table 5). We use marginal metric score based channel allocation method as it gives the best performance which we established in section VI-A and VI-C. We evaluate the performance of different CU pattern driven channel allocation feature of the proposed Algorithm 2 by comparing it relative to channel allocation where this reallocation feature is not utilized (i.e. steps 7 to 11 in Algorithm 2 are ignored).

Table 5 shows the gain in the average sum metric obtained for our CAD predictions based channel allocation relative to the algorithm with no reaction to the different CU patterns.

TABLE 5. Percentage gain in average sum metric when utilizing the proposed algorithm.

Scenario	Prediction based allocation	Prediction based allocation with CAD module	Performance gain
Case I: One different CU pattern, 4 hrs	278.03 Mbps	290.94 Mbps	5.12%
Case II: Two different CU patterns, 4 hrs and 3 hrs	331.57 Mbps	358.90 Mbps	9.06%

In Table 5, it can be seen that the proposed CAD predictions based channel allocation method which takes into account different CU patterns achieves significant performance gains over the algorithm which does not take into account different CU patterns for reallocation. The gain in the average sum metric can be explained as follows. When a certain AP in a channel has high CU demand which cannot be satisfied in that channel, it results in a reduced metric score of itself and other APs in the same channel. The proposed CAD predictions based channel allocation algorithm detects this behavior and moves the affected AP to a channel which can better satisfy its CU demand by performing channel reallocation. This improves its own metric score and the metric scores of other APs in the previous channel which helps to increase the overall sum metric. It is also worth noting that our algorithm either performs equally well or it improves the performance of a WLAN over channel allocation which does not take into account different CU patterns. Hence, there is no penalty in performance in existing systems by incorporating the proposed algorithm.

VII. CONCLUSION

In this paper, we have presented an uncertainty-aware DL model for robust prediction of wireless CU and real-time change detection in CU distribution. To account for the uncertainty in the DL model, we have used an encoder-decoder framework based DL model using BNNs. Our results have shown that the prediction performance of the proposed model is as good as other models. However, in addition to predictions, an additional feature of our proposed model is that it can consistently quantify the uncertainty in predictions using PIs. By using computed PIs, we have shown that we can perform change detection in CU distribution accurately. We have also developed a channel allocation algorithm for WLAN called *ProReact* which utilizes the predictions from the DL model to compute a novel metric score which is used to find efficient channel allocation plan for the APs in the network. Moreover, the proposed algorithm also utilizes the change detection in CU feature of the proposed DL model to perform channel reallocation when the CU distribution changes. Our results have shown that our channel allocation algorithm achieves fast convergence leading to high sum rates in the network.

One possible extension of our work we envisioned for is to address the problem of coexistence of the licensed assisted access of future wireless mobile networks and WiFi with the application of the proposed CAD predictions based channel allocation algorithm.

REFERENCES

[1] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions," *IEEE Access*, vol. 7, pp. 137184–137206, 2019.
[2] C.-X. Wang, M. D. Renzo, S. Stanczak, S. Wang, and E. G. Larsson, "Artificial intelligence enabled wireless networking for 5G and beyond: Recent advances and future challenges," *IEEE Wireless Commun.*, vol. 27, no. 1, pp. 16–23, Feb. 2020.
[3] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. Emad Ul Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.
[4] I. P. Chochliouros, A. Kostopoulos, M. Payaró, C. Verikoukis, S. D. C. di Vimercati, E. Vinogradov, V. Ranjbar, J. Vardakas, M. A. Rahman, P. Soumplis, and E. Varvarigos, "Machine learning-based, networking and computing infrastructure resource management," in *Proc. Artif. Intell. Appl. Innov. Workshops (AIAI)*, I. Maglogiannis, J. Macintyre, and L. Iliadis, Eds. Cham, Switzerland: Springer, 2021, pp. 85–94.
[5] E. Castañeda, A. Silva, A. Gameiro, and M. Kountouris, "An overview on resource allocation techniques for multi-user MIMO systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 239–284, 1st Quart., 2017.
[6] Cisco Systems. *Cisco Meraki Best Practice Design*. Accessed: Sep. 18, 2020. [Online]. Available: https://documentation.meraki.com/Architectures_and_Best_Practices/Cisco%_Meraki_Best_Practice_Design
[7] W. Kim, S. Ryu, J. Won-Ki Hong, and Y.-J. Suh, "WLANMan: A cloud-based wireless LAN management system in ONOS controllers," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2016, pp. 774–775.
[8] J. Guo and C. Yang, "Predictive resource allocation with deep learning," in *Proc. IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, Aug. 2018, pp. 1–7.
[9] Q. Guo, R. Gu, Z. Wang, T. Zhao, Y. Ji, J. Kong, R. Gour, and J. P. Jue, "Proactive dynamic network slicing with deep learning based short-term traffic prediction for 5G transport network," in *Proc. Opt. Fiber Commun. Conf. (OFC)*, Mar. 2019, pp. 1–3.
[10] P. Yu, F. Zhou, X. Zhang, X. Qiu, M. Kadoch, and M. Cheriet, "Deep learning-based resource allocation for 5G broadband TV service," *IEEE Trans. Broadcast.*, vol. 66, no. 4, pp. 800–813, Dec. 2020.
[11] L. Zhu and N. Laptev, "Deep and confident prediction for time series at uber," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2017, pp. 103–110.
[12] L. V. Jospin, W. L. Buntine, F. Boussaïd, H. Laga, and M. Bennamoun, "Hands-on Bayesian neural networks—A tutorial for deep learning users," *CoRR*, vol. abs/2007.06823, 2020. [Online]. Available: https://arxiv.org/abs/2007.06823
[13] Z. Khan and J. J. Lehtomäki, "FPGA-assisted real-time RF wireless data analytics system: Design, implementation, and statistical analyses," *IEEE Access*, vol. 8, pp. 4383–4396, 2020.
[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org
[15] H. Zhang, H. Zhang, K. Long, and G. K. Karagiannidis, "Deep learning based radio resource management in NOMA networks: User association, subchannel and power allocation," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2406–2415, Oct. 2020.
[16] S.-M. Tseng, Y.-F. Chen, C.-S. Tsai, and W.-D. Tsai, "Deep-Learning-Aided cross-layer resource allocation of OFDMA/NOMA video communication systems," *IEEE Access*, vol. 7, pp. 157730–157740, 2019.
[17] S. K. Padaganur and J. D. Mallapur, "A neural network based resource allocation scheme for 4G LTE heterogeneous network," in *Proc. 2nd Int. Conf. for Converg. Technol. (I2CT)*, Apr. 2017, pp. 250–254.
[18] Z. Xu, J. Guo, and C. Yang, "Predictive resource allocation with interference coordination by deep learning," in *Proc. 11th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2019, pp. 1–6.
[19] K. I. Ahmed, H. Tabassum, and E. Hossain, "Deep learning for radio resource allocation in multi-cell networks," *IEEE Netw.*, vol. 33, no. 6, pp. 188–195, Nov./Dec. 2019.

[20] Z. Khan, J. Lehtomäki, C. Ganewattha, and S. Shahabuddin, "Histograms to quantify dataset shift for spectrum data analytics: A SoC based device perspective," in *Proc. 2nd 6G Wireless Summit (6G SUMMIT)*, Mar. 2020, pp. 1–5.
[21] A. Subbaswamy, P. Schulam, and S. Saria, "Preventing failures due to dataset shift: Learning predictive models that transport," in *Proc. 22nd Int. Conf. Artif. Intell. Statist. (AISTATS)*, Apr. 2019, pp. 3118–3127. [Online]. Available: http://proceedings.mlr.press/v89/subbaswamy19a/subbaswamy19a.pdf
[22] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2009.
[23] P. Z. Jin and C. H. S. Hoi, "OTL: A framework of online transfer learning," in *Proc. 27th Int. Conf. Mach. Learn., (ICML)*, 2010, pp. 1231–1238.
[24] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, New York, NY, USA, vol. 48, Jun. 2016, pp. 1050–1059.
[25] S. Khan, A. S. Al-Mogren, and M. F. Alajmi, "Using cloud computing to improve network operations and management," in *Proc. 5th Nat. Symp. Inf. Technol., Towards New Smart World (NSITNSW)*, Feb. 2015, pp. 1–6.
[26] N. Zhang, N. Cheng, A. T. Gamage, K. Zhang, J. W. Mark, and X. Shen, "Cloud assisted HetNets toward 5G wireless networks," *IEEE Commun. Mag.*, vol. 53, no. 6, pp. 59–65, Jun. 2015.
[27] A. Bhartia, B. Chen, F. Wang, D. Pallas, R. Musaloiu-E, T. T.-T. Lai, and H. Ma, "Measurement-based, practical techniques to improve 802.11ac performance," in *Proc. Internet Meas. Conf.*, Nov. 2017, pp. 205–219.
[28] Z. Khan, J. J. Lehtomäki, R. Aguilar-Gonzalez, R. Vuohtoniemi, E. Hossain, L. A. DaSilva, and A. Marshall, "Database-assisted distributed and cloud-based access methods for unlicensed and radar bands," *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 3, pp. 404–419, Sep. 2017.
[29] J. R. Marden and A. Wierman, "Distributed welfare games," *Oper. Res.*, vol. 61, no. 1, pp. 155–168, 2013.
[30] J. R. Marden and A. Wierman, "Overcoming limitations of game-theoretic distributed control," in *Proc. 48h IEEE Conf. Decis. Control (CDC) Held Jointly With 28th Chin. Control Conf.*, Dec. 2009, pp. 6466–6471.

CHANAKA GANEWATTHA received the B.Sc. degree in electronics and telecommunication engineering from the University of Moratuwa, Sri Lanka, in 2016, and the M.Sc. degree in electrical engineering from the University of Oulu, Finland, in 2019, where he is currently pursuing the Ph.D. degree in communications engineering.

From 2016 to 2018, he was an Electronic Engineer with Paraqum Technologies, Sri Lanka. His research interests include FPGA based digital hardware implementations, software-defined radio, algorithms in signal processing, and machine learning.

ZAHEER KHAN received the M.Sc. degree in electrical engineering from the University College Borås, Sweden, in 2007, and the Dr.Sc. degree in electrical engineering from the University of Oulu, Finland, in 2011. He has worked as a Tenure Track Lecturer at the University of Liverpool, Liverpool, U.K., from 2016 to 2017, and a Research Fellow/Principal Investigator at the University of Oulu, from 2011 to 2016. He is currently an Adjunct Professor at the University of Oulu. His research interests include implementation of advanced signal processing and wireless communications algorithms on Xilinx FPGAs and Zynq System-on-Chip (SoC) boards, application of game theory to model distributed wireless networks, prototyping access protocols for wireless networks, the IoT location tracking systems, cognitive and cooperative communications, and wireless signal design. He was a recipient of the Marie Curie Fellowship, from 2007 to 2008.

MATTI LATVA-AHO (Senior Member, IEEE) received the M.Sc. degree in electrical engineering and the Lic.Tech. and Dr.Tech. degrees from the University of Oulu, Finland, in 1992, 1996, and 1998, respectively. From 1992 to 1993, he was a Research Engineer at the CDMA Systems Research Group, Nokia Mobile Phones. From 1994 to 1998, he was a Research Scientist at the Telecommunication Laboratory and the Centre for Wireless Communications (CWC), University of Oulu. After graduation he became the Director of CWC, in 1998, and the Head of the Department of Communications Engineering, in 2011.

JANNE J. LEHTOMÄKI (Member, IEEE) received the Ph.D. degree from the University of Oulu, Finland, in 2005. He was a Visiting Scholar at the Georgia Tech, Atlanta, GA, USA, in Fall 2013 Semester. He is currently an Adjunct Professor at the Centre for Wireless Communications, University of Oulu. He is focusing on spectrum measurements and terahertz band wireless communication. He was an Editorial Board Member of physical communication. His coauthored paper receiving the Best Paper Award in IEEE WCNC 2012. He has served as a Guest Associate Editor for the *IEICE Transactions on Communications* Special Section, from February 2014 to July 2017, and a Managing Guest Editor for *Nano Communication Networks* Special Issue, in June 2016. He was the General Co-Chair of IEEE WCNC 2017 International Workshop on Smart Spectrum, the TPC Co-Chair of IEEE WCNC 2015 and 2016 International Workshop on Smart Spectrum, and the Publicity/Publications Co-Chair of ACM NANOCOM 2015, 2016, and 2017.