Coverage Probability Prediction in Cellular Networks Based on Spatial Distribution of Base Stations

Arindam Ghosh
Electrical and Computer Engineering
Carnegie Mellon Univerisity
Pittsburgh, PA 15213
arindam2@andrew.cmu.edu

Phalguna Dasaratha Mankar
Electrical and Computer Engineering
Carnegie Mellon Univerisity
Pittsburgh, PA 15213
pdasarat@andrew.cmu.edu

1 Introduction

The performance of a wireless network is often studied either by extensive simulation or by using simplified mathematical models such as random wireless networks [3] that try to approximate the real world network deployment. However, both approaches have their shortfalls. Simulations require huge computational resources and the results of that particular simulation will not be applicable to any other network topology. The analytical models on the other hand fall short due to the simplified assumptions within which cannot be confidently applied to a real life scenario. Even with such simplifications, the analytical tractability is not enough to obtain many desired expressions in computable forms. Further, the results that could be computed are averaged out over all the possible spatial configurations of the base stations (BSs), considering an infinite field. Clearly then, we cannot blindly use the averaged-out results to predict the performance of a particular snapshot of a real life network.

Thus, one would desire a model which could predict the performance of a network based on the actual locations of the BSs and the network parameters (base station intensity λ, path loss exponent α, transmit power). From a network operator's point of view, one would like to know that if the base stations are placed in so and so locations, and the parameters are used with so and so values how will the network perform. Through this study, our aim is to provide such a model using the data driven approach of neural networks that would predict the coverage probability of a typical user (considered to be at the origin in this particular study) using the spatial information of these base stations and the network parameters.

2 Literature

In the field of wireless communication, there has been a large interest in neural networks due to its ability to capture the real complexity of the network when mapping various network parameters to output feature (any metric that captures the network performance)[1-2]. In [4], the authors propose a neural network based approach for a better handover decision in heterogeneous networks. In [5], the authors propose a distributed deep neural network to learn the optimal power allocation for a device-to-device network. The problem statement of our study, however, finds its motivation from [6] where fully connected Neural Network (FC-NN) [6, page-6] is used to map the network parameters (such as λ, α, transmit power) into the coverage probability. In that study, however, the coverage probability is computed only as a function of the network parameters: path-loss exponent α, Fading (Nakagami) parameters (h), BS density λ, Signal Power P etc., and have averaged out the effect of the spatial distributions of the BSs. This, therefore, ignores the impact of distinct spatial features of the network on the coverage probability. In our study, we aim to take particularly the spatial information of the BSs into account while predicting the coverage probability.

10807 Class Project (2016).

3 Problem Description

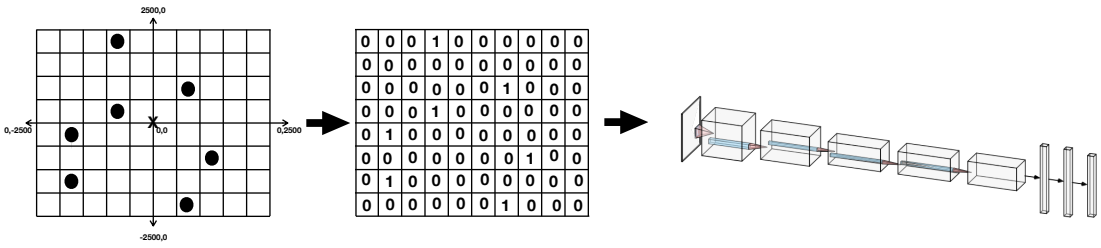

Figure 1: System Model

3.1 Cellular Network Architecture

We consider a cellular network on the 2D plane of dimension $5\text{Km}\times\ 5\text{Km}$ wherein the base stations (BSs) are distributed as shown by the black circles in Fig. 1. We choose the typical user (who's coverage probability we are interested in) to be at the origin (marked as cross in the figure). We assume the serving BS to typical user association scheme to be Nearest Base station Association Scheme (NBAS) i.e. the user is served by the nearest base station. Let b_0 denote the BS serving the typical user; then, all other base stations $b \neq b_0$ act as interferers to the user. Under such a setting, and considering the down-link channel, the SINR experienced by the typical user is then given by [6, page-3]

$$\text{SINR} = \frac{P h_{b0} d_{b0}^{-\alpha}}{\sigma^2 + \sum_{b \neq b_0} P h_b d_b^{-\alpha}} \tag{1}$$

where P is the transmit power of a BS at location b, d_b is the distance between the typical user and BS at b, α is the path loss exponent, $h_b \sim \exp(1)$ is the exponentially distributed small-scale fading power gain.

For this setup, our aim is to predict the coverage probability P_c which is popularly defined, in the literature and [6, page-3], as the probability that the SINR is above a given threshold τ , i.e.

$$P_c(\tau) = \mathbb{P}(\text{SINR} > \tau) \tag{2}$$

3.2 Prediction Problem

Based on the probability $P_c(\tau)$, in (2), we define the 10 classes of coverage (or network performance). A particular network-snapshot (i.e. with a particular spatial distribution of BSs) is then assigned the class label y as follows:

$$y = \begin{cases} 0 & \text{if } P_c \in (0.9, 1] \\ 1 & \text{if } P_c \in (0.8, 0.9] \\ 2 & \text{if } P_c \in (0.7, 0.8] \\ \vdots & \vdots \\ 9 & \text{if } P_c \in [0, 0.1] \end{cases} \tag{3}$$

Our task is to predict which class the network belong to (given the snapshot of the network with BS distribution) using just the location of the BSs (or equivalently using their distances from the observation point o). This is a quite a challenging task as $P_c(\tau)$ is an extremely complicated function of the BS distances $\{d_b\}$. Notice that, for (1), not only does it involve the mapping of the distances $\{d_b\}$, but also requires figuring out the serving BS (i.e. the closest BS at distance d_{b0} from o). The identity of the serving BS basically determines what goes into the numerator and it's absence determines the denominator. Then, Eq. (2) requires computing the distribution of SINR so to get the probability $\mathbb{P}(\text{SINR} > \tau)$ in (2). Finally, the overall mapping gets complicated by the thresholding in (3).

Note that in [6, page-5], different to our contribution, the aim was to predict $P_c(\tau)$ just as a function of network parameters: path-loss exponent α, Fading (Nakagami) parameters (h), BS density λ,

Signal Power P etc., and averaged out the effect of the spatial distributions of the BSs. They have not considered the effect of individual BS distances $\{d_b\}$ on $P_c(\tau)$, which our study considers.

4 Proposed Solution

We approach the problem in two ways:

Classification Formulation:
We directly treat it as a classification problem and use FC-NN to predict the y of the test network snapshot. We use cross-entropy error as the loss function.

Regression Formulation: Here, we use FC-NN to do regression and try to predict the exact value of $P_c(\tau)$ and then assign the class y based on the rule in (3). We use Root-Mean-Square Error (RMSE) as the loss function.

Data Pre-processing to be applicable to Neural Networks: We start with converting a particular spatial snapshot (shown in Fig. 1 at the leftmost side) of base stations, denoted by S_n (for $1 \leq n \leq N$, where N could denote the number of training snapshots), into a format that can be fed into the neural network. To accomplish this, we divide the 2-D square region into an $m \times m$ grid (in Fig. 1 $m = 8$ as an example). From this grid, we construct an $m \times m$ matrix X_n containing 0's or 1's (shown in Fig. 1 as the second grid from left side) as following: for a particular grid box indexed i, j, we set the element $X_{n,i,j} = 1$ if a base station is present in that box, otherwise we set $X_{n,i,j} = 0$. Then the input to the neural network is simply the $m \times m$ matrix X_n (or the flattened version of X_n when using only FC-NN architecture).

The task of the neural network is then to detect and learn the spatial arrangement of these 1's and 0's in X_n and associate them with a particular class y_n at the output. By such a formulation of the problem, learning of the spatial pattern of 1's and 0's becomes equivalent to learning the spatial distribution of the base stations in the actual 2-D network when predicting coverage probability.

5 Performance Evaluation and Results

5.1 Data Set

We use $N = 10,000$ training samples, $V = 5,000$ validation samples, and $T = 5,000$ testing samples. As it is difficult to get real-deployment based data from network operators, we use Monte Carlo simulations to generate our dataset. We generate $N + V + T$ network snapshots (i.e. spatial configuration of the base stations), where, in each snapshot (X_n) the base stations are placed randomly. Then, for each X_n, we obtain the true label y_n by simulating $10,000$ realizations of that particular X_n (Note that for all the realizations of X_n, although the spatial locations are fixed, they differ from each other based on the random fading gains of the communication channels). Therefore, here we are using Monte Carlo simulations to compute whether SINR $> \tau$ or not for the $10,000$ realizations and then averaging to get the P_c and consequently the y_n for X_n. We repeat this for all $N + V + T$ network snapshots to get our final dataset.

5.2 Neural Network Architecture

Classification:

Input –> Multiple FC Layers (Sigmoid) –> 10 Output Nodes (Softmax)

Regression:

Input –> Multiple FC Layers (ReLU) –> 1 Output Node (Sigmoid)

5.3 Baseline Performance (Classification Formulation)

As baseline, we study the accuracy of prediction of the coverage probability using a two layer FC-NN which consists of 100 units each in both the layers. At the input, we feed a flattened version of X_n (original size $m \times m$, with $m = 50$) to this FC-NN and at the output we get the label y. We employ cross entropy error as the loss function.

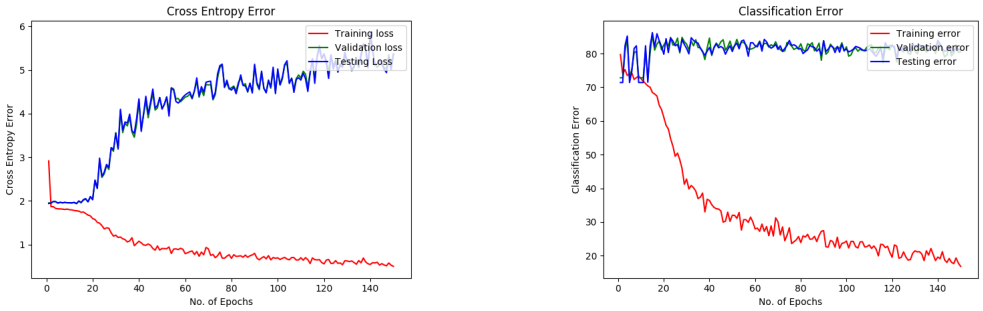

Figure 2: Cross-entropy loss (left), Classification error (right), Fully connected MLP with two hidden layers: $H1 = 100$, $H2 = 100$, Learning rate $\eta = 0.01$, Momentum $\beta = 0.5$

From the above figures, we observe that the FC-NN quickly overfits the training data. this is because we had earlier used only 2000 training inputs, 1000 validation inputs and 1000 testing inputs.

5.4 Classification Formulation

5.4.1 Optimal Depth of the Neural Network

Here, we decide on the optimum number of hidden layers to be used that work best for the classification problem at hand.

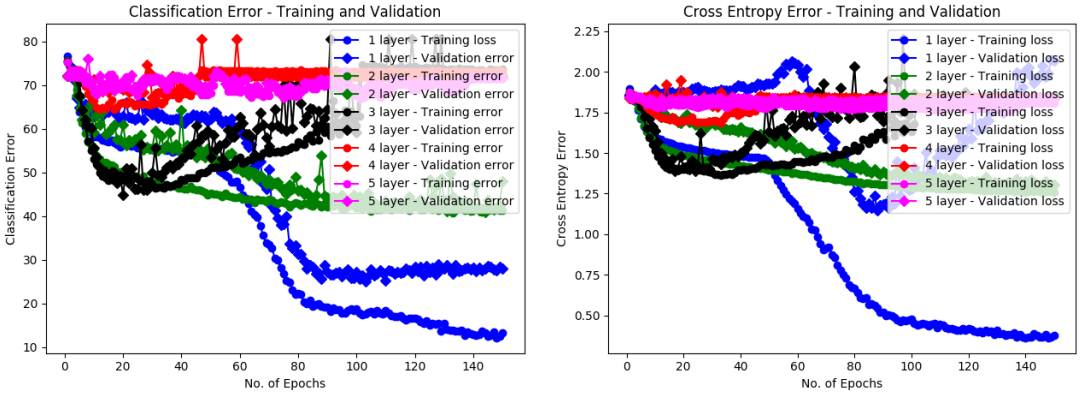

Figure 3: FC-NN with 1-5 hidden layers (each with 36 units), Learning rate $\eta = 0.01$, Momentum $\beta = 0.5$

From Fig. 3, we see that the FC-NN works better for 1 and 2 hidden layers. The performance when no. of layers = 3,4, and 5 are significantly worse. Therefore, we restrict ourselves to 1 and 2 layer FC-NN for further evaluation. Further, Fig. 3, we find that the optimum number of epochs = 80 for early stopping.

5.4.2 Optimal Number of Hidden Units

From the training and test performance in Fig. 4 and Fig. 4, we see that the better performing candidate architectures are the 1-layer (36 units) and 2-layer (each layer 36 units and each layer 16 units). Our further evaluations will be using these three candidates.

Note that in these figures, learning rate is halved after 50 epochs.

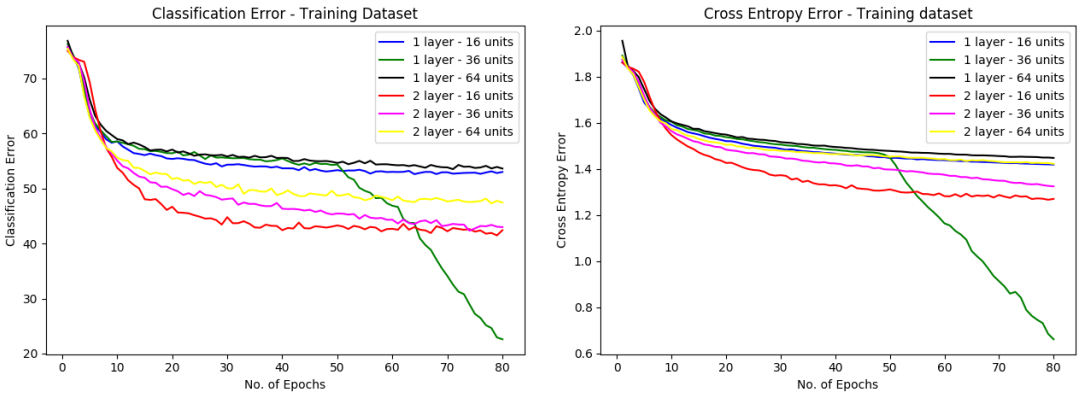

Figure 4: Training behaviour for different no. of hidden units. Learning rate $\eta = 0.01$ (halved after 50 epochs), Momentum $\beta = 0.5$.

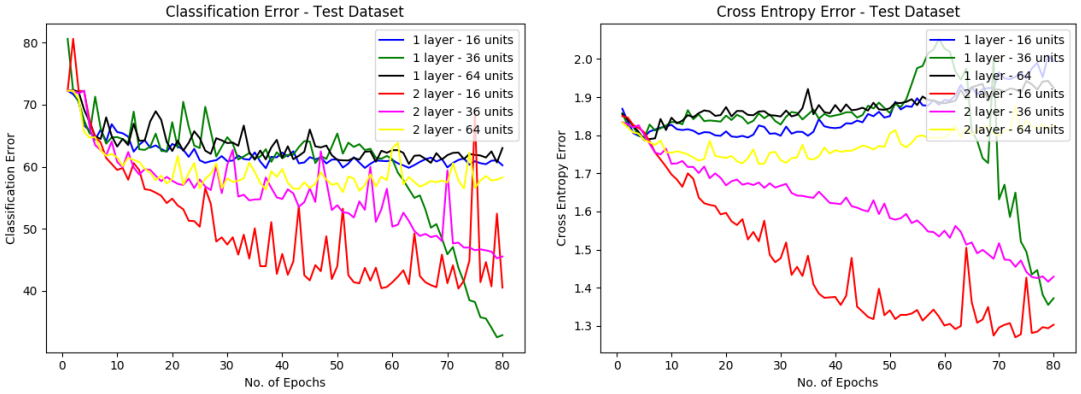

Figure 5: Test behaviour for different no. of hidden units. Learning rate $\eta = 0.01$ (halved after 50 epochs), Momentum $\beta = 0.5$.

5.4.3 Effect of Batch Normalization

In Fig. 6, we see that batch normalization helps in achieving much better training and testing accuracy. On comparison, we observe that 1-later with 36 units outperform the other two candidates.

5.4.4 Effect of Weight Decay

As seen from Fig. 7, it doesn't have a drastic change in the performance. Thus, for our final model and evaluation, we keep weight decay along with batch normalization.

5.4.5 Optimal Performance of Classification

Concluding the analysis, we see that for the classification formulation of the problem, we get the best performance with 1-layer 36 units FC-NN (with batch normalization and weight decay), which gives $\approx 75\%$ accuracy on the test data.

5.5 Regression Formulation:

5.5.1 Optimal Depth and Number of Hidden Units

From the Table below, we see that, for a single layer architecture, adding more number of units don't help much.

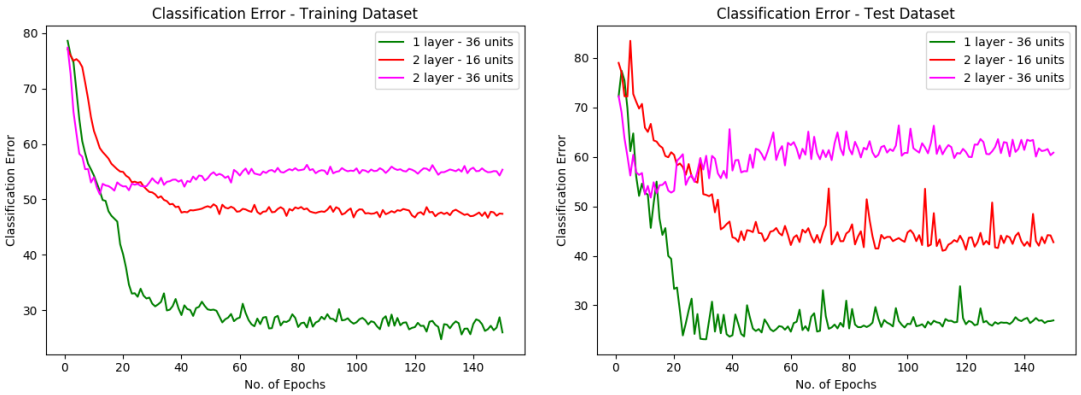

Figure 6: Effect of Batch normalization on training (left) and testing (right). Learning rate $\eta = 0.01$ (halved after 50 epochs), Momentum $\beta = 0.5$.

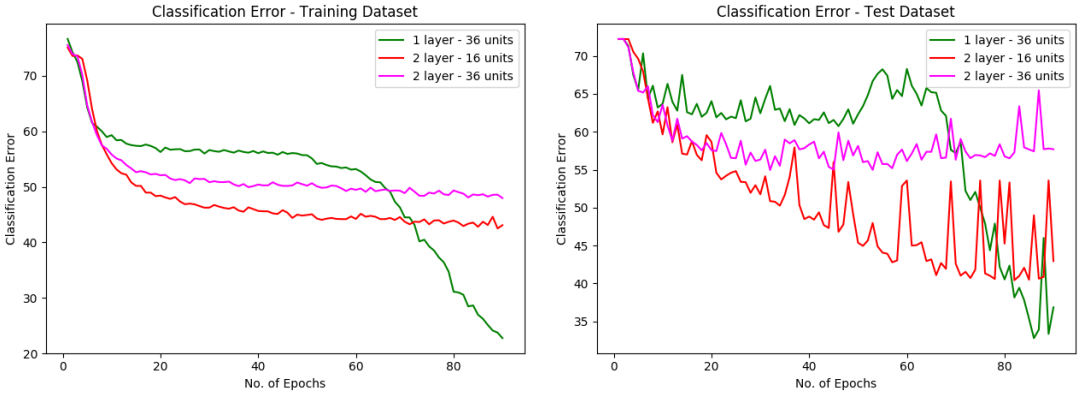

Figure 7: Effect of Weight Decay on training (left) and testing (right). Learning rate $\eta = 0.01$ (halved after 50 epochs), Momentum $\beta = 0.5$.

# Layers	# Units (in each layer)	Epochs	Test Accuracy
1	20	50	16.02 %
1	20	100	15.40 %
1	50	50	15.78 %
1	50	100	15.08 %
1	100	50	13.48 %
1	100	100	15.00 %
2	20	50	19.92 %

However, it is also evident that adding more layers (# layers = 2) is beneficial. Therefore, we study the effect of increasing the number of hidden units by first considering 2 layers in the table below.

# Layers	# Units (in each layer)	Epochs	Test Accuracy
2	20	50	19.92 %
2	20	100	30.46 %
2	20	150	24.68 %
2	50	100	36.78 %
2	100	100	40.78 %
2	200	100	42.02 %
2	500	100	38.80 %

From the table above, we find that adding more units definitely increase the accuracy, but only upto 200 units. The architecture with 500 units has a decreased performance. Thus, we select 200 units to

be the optimal no. of units at this point. Further, we find that stopping at 100 epoch is a good choice for early stopping criterion.

Next, we evaluate the performance by increasing the no. of layers. Observe, in the table below, that, for 200 hidden units, the performance improves from 2-layer to 3-layer but decreases for 4-layers. Therefore, we next decrease the no. of hidden units for 4-layer to 100 (at each layer) and see the accuracy jumps to 48.16%.

# Layers	# Units (in each layer)	Epochs	Test Accuracy
2	200	100	42.02 %
3	200	100	43.62 %
4	200	100	42.28 %
4	100	100	48.16 %
5	100	100	51.32 %
6	100	100	50.00 %
10	50	100	55.66 %

Next, we observe that with 100 hidden units increasing depth to 6 doesn't improve the performance. This calls for a reduction in the number of hidden units as we increase the depth of the network. As observed, the above strategy seems to work as for 10 FC layers with 50 units each we again see an improvement jump to 55%.

5.5.2 Optimal Performance of Regression

In conclusion, we observe that the regression problem is more difficult to solve and requires more network resources (more no. of layers and units) as it only achieves a maximum accuracy of 55 %.

6 Conclusion

Through this study, our aim is to predict the coverage probability of a typical user inna cellular network using the spatial information of the base stations. We convert this problem into a form that can be fed into neural networks by creating a matrix of 1s and 0s which basically capture the BS locations in a square 2D grid. Then, we approach the problem in two way: (a) by directly formulating it as a classification problem and (b) by formulating it as a regression problem then classifying at the end. On these formulations, we study the performance of various architectures and hyper-parameters of neural network. Through our experiments, we find that the regression formulation (achieves a maximum accuracy of 55 %) is much harder to solve and requires a lot of network resources (i.e. more depth and length of the neural network). Whereas, the classification formulation performs lot better just with 1 layer FC-NN with 36 units. Thus, the classification formulation and the neural network architecture proposed (with the given hyper-parameter settings) should be used to solve the given prediction problem and others related to it.

7 References and citations

[1] C. Zhang, P. Patras and H. Haddadi, "Deep Learning in Mobile and Wireless Networking: A Survey," in IEEE Communications Surveys & Tutorials.

[2] P. Casas, "Machine learning models for wireless network monitoring and analysis," 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Barcelona, 2018, pp. 242-247.

[3] J. G. Andrews, F. Baccelli and R. K. Ganti, "A Tractable Approach to Coverage and Rate in Cellular Networks," in IEEE Transactions on Communications, vol. 59, no. 11, pp. 3122-3134, November 2011.

[4] N. Alotaibi and S. Alwakeel, "A neural network based handover management strategy for heterogeneous networks," in International Conference on Machine Learning and Applications (ICMLA), 2015, pp. 1210–1214.

[5] J. Kim, J Park, J. Noh, and S. Cho, "Completely distributed power allocation using deep neural network for device to device communication underlaying LTE," arXiv preprint arXiv:1802.02736, 2018.

[6] El Hammouti, H, Ghogho, M and Zaidi, SAR (Accepted: 2018) A Machine Learning Approach to Predicting Coverage in Random Wireless Networks. In: GLOBECOM 2018: IEEE Global Communications Conference.