Graph Convolutional Neural Networks for Optimal Load Shedding under Line Contingency

Cheolmin Kim
Department of Industrial Engineering and Management Sciences
Northwestern University, Evanston, IL, USA
Email: cheolminkim2019@u.northwestern.edu

Kibaek Kim
Prasanna Balaprakash
Mihai Anitescu
Mathematics and Computer Science Division
Argonne National Laboratory, Lemont, IL, USA
Email: {kimk, pbalapra, anitescu}@anl.gov

***Abstract*—Power system operations under contingency need to solve large-scale complex nonlinear optimization problems in a short amount of time, if not real time. Such nonlinear programs are computationally challenging and time-consuming and thus do not scale with the size of the power system network. We apply a graph convolutional network (GCN) model, as a supervised learning model, for predicting an optimal load-shedding ratio that prevents transmission lines from being overloaded under line contingency (i.e., line tripping). In particular, we exploit the power system network topology in the GCN model, where the topology information is convoluted over the neural network. Using IEEE test cases, we benchmark our GCN model against a classical neural network model and a linear regression model and show that the GCN model outperforms the others by an order of magnitude.**

***Index Terms*—Graph covolutional network, neural network, machine learning, alternating current power system, contingency analysis.**

I. Introduction

Power grid operations involve a variety of decision-making problems (e.g., unit commitment, optimal power flow, economic dispatch) that can be formulated as optimization problems. By using off-the-shelf optimization solvers, one can find optimal solutions for such decision-making problems in a reasonable amount of time. Despite the solution quality the optimization-based approach provides, however, it may not be appropriate for timely decision-making tasks (e.g., security- or emergency-related tasks), because power grid operation problems are too complex to be solved in real time.

Load shedding is important in power system operations particularly under contingency events (e.g., line failure [1]). When one or more transmission lines are tripped, the system needs to correct the stability by adjusting power dispatch, including voltage angles and magnitudes, generation, and load. Shedding load is necessary when the system ramp capacity is pushed to the limit. Failure to shed sufficient load from the system can lead to some transmission lines being overloaded and additional line failures, possibly resulting in cascading failures. On the other hand, undesired load losses can occur if more load is shed than necessary. Therefore, finding the optimal amount of load shedding that minimizes load losses but also prevents additional line outages is an important issue for maintaining the stability and efficiency of power systems.

One way to obtain the optimal amount of load shedding is to solve the alternating current optimal power flow (ACOPF) problem with respect to minimizing the total amount of load shedding [1]. The ACOPF problem is challenging, however, because of its nonlinearity and nonconvexity. Therefore, its solution is hard to be obtained in a short time and does not scale with the size of the power system. By training a machine learning model with high-quality training data (possibly a collection of outputs from an optimization model for a variety of scenarios), one can build a model that returns a high-quality solution in real time. In order to mimic the complex mathematical operations involved in solving a complex optimization problem, neural network models have been proposed and proved effective for various power system applications (e.g., [2], [3], [4], [5]).

In this paper, we introduce the first application of a graph convolutional network (GCN) [6] to power grid operations. In particular, we develop a deep learning framework that implements a GCN model as a supervised learning model to predict an optimal load-shedding amount in order to avoid subsequent transmission line overloads in the power system. The GCN model generates intermediate features utilizing the adjacent matrix of the underlying power system network topology, making it a better choice for capturing interactions on a given graph, compared with a classical neural network model that combines input features without taking the underlying topology information into account. To apply GCN to load-shedding operations, we develop a variant of the ACOPF model that minimizes the load-shedding ratio given a load profile and a contingency scenario. For a variety of demand profiles and contingency scenarios, the optimization model is run to generate data samples, which are then used to train our GCN model to predict an optimal load-shedding ratio for a given system state. To demonstrate the effectiveness of our GCN model, we provide extensive numerical experiments on IEEE bus systems, where we compare the performance of our GCN model with a classical neural network model and a linear regression model.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357.

II. Optimal Load-Shedding Model

In this section, we describe the optimal load-shedding model that we use to obtain the optimal shedding ratio for each demand profile and contingency scenario. We start with the notation used throughout this paper.

A. Notation

A power grid consists of a set of buses $\mathbb{N}$ and a set of lines $\mathbb{L}$ that connect buses. Among the buses is a set of generators $\mathbb{G}$ that generate power. Also included is a set of reference buses $\mathbb{N}_{\text{ref}} \subset \mathbb{N}$ that provide an angular reference to other buses. For each entity, we define related physical quantities as follows.

1) Bus: For each bus $i \in \mathbb{N}$,

- $V_m[i]/V_a[i]$: voltage magnitude/angle (p.u./degrees)
- $P_d[i]/Q_d[i]$: real/reactive power demand (MW/MVAr)
- $G_s[i]/B_s[i]$: shunt conductance/susceptance (MW/MVAr demanded/injected at $V = 1.0$ p.u.)
- $V_m^{\min}[i]/V_m^{\max}[i]$: min/max voltage magnitude (p.u.)

2) Line: For each line $l \in \mathbb{L}$,

- $r[l], x[l]$: resistance/reactance (p.u.)
- $b[l]$: total line charging susceptance (p.u.)
- $\tau[l]$: transformer off nominal turns ratio
- $\theta[l]$: transformer phase shift angle (degrees)
- $\bar{f}[l]$: apparent power limit on line l

3) Generator: For each generator $k \in \mathbb{G} \subset \mathbb{N}$,

- $P_g[k]/Q_g[k]$: real/reactive power output (MW/MVAr)
- $P_g^{\min}[k]/P_g^{\max}[k]$: min/max active power output (MW)
- $Q_g^{\min}[k]/Q_g^{\max}[k]$: min/max reactive power output (MVAr)
- $a[k]$: participation factor of generator k in real power contingency response (1)

4) Reference Bus: For each reference bus $i_{\text{ref}} \in \mathbb{N}_{\text{ref}}$,

- $V_a^{\text{ref}}[i_{\text{ref}}]$: voltage reference angle (degrees)

B. Formulation

In this model, we assume that line contingency occurs after the power dispatch is set to meet a load profile $D = (P_d, Q_d)$. Let $X = (P_g, Q_g, V_m, V_a)$ be a dispatch solution, and let $\mathbb{L}' \subset \mathbb{L}$ be a set of active lines representing a contingency scenario. To find the optimal load-shedding ratio that minimizes load losses and maintains the stability of system (i.e., feasibility), we consider the following optimization model.

1) Decision Variables: The decision variables are

- $0 \le \rho \le 1$: load-shedding ratio
- P_g', Q_g', V_m', V_a': power dispatch after line contingency
- $-1 \le P_g^{\Delta} \le 1$: generator participation factor for real power contingency response (p.u.)
- $0 \le f_s[l']$: slack for power overflow on each line $l' \in \mathbb{L}'$.

2) Objective Function: The objective function to be minimized is

$$\rho + \lambda \sum_{l' \in \mathbb{L}'} f_s[l'].$$

It comprises the load-shedding ratio ρ and the sum of the power overflow $f_s[l']$ on each line $l' \in \mathbb{L}'$, where their relative weights are governed by the penalty parameter λ.

3) Generation Constraints: For each generator $k \in \mathbb{G}$, we have real/reactive power generation limits:

$$P_g^{\min}[k] \le P_g'[k] \le P_g^{\max}[k],$$
$$Q_g^{\min}[k] \le Q_g'[k] \le Q_g^{\max}[k].$$

In addition to these constraints, real power generation has the following extra constraint for each generator $k \in \mathbb{G}$:

$$P_g'[k] = P_g[k] + a[k]P_g^{\Delta},$$

which enforces that the change of real power output $P_g'[k] - P_g[k]$ before and after line contingency should be proportional to prescribed participation factors $a[k]$.

4) Voltage Constraints: Each bus $i \in \mathbb{N}$ has a box constraint on the voltage magnitude $V_m'[i]$:

$$V_m^{\min}[i] \le V_m'[i] \le V_m^{\max}[i].$$

Voltage angles are fixed for each reference bus $i_{\text{ref}} \in \mathbb{N}_{\text{ref}}$:

$$V_a'[i_{\text{ref}}] = V_a^{\text{ref}}[i_{\text{ref}}].$$

5) Line Flow Constraints: Each line $l' = (f, t) \in \mathbb{L}'$ has a soft limit on power flow $S[f, t]$ and $S[t, f]$:

$$\max(|S[f,t]|, |S[t,f]|) \le \bar{f}[l'] + f_s[l'].$$

As expressed above, power overflow is allowed but penalized in the objective function by the amount it exceeds the limit.

6) AC Power Flow Equations: Let $\mathbb{G}_i$ be the set of generators located at bus i, and let $Y = G + jB$ be the admittance matrix of the power system after line contingency, where G and B are real matrices and $j = \sqrt{-1}$. For each bus $i \in N$, we have

$$\sum_{k_i \in \mathbb{G}_i} P_g[k_i] - (1-\rho)P_d[i]$$
$$= \sum_{k=1}^{n} V_m[i]V_m[k]\big(G_{ik}\cos\big(V_a[ik]\big) + B_{ik}\sin\big(V_a[ik]\big)\big)$$

and

$$\sum_{k_i \in \mathbb{G}_i} Q_g[k_i] - (1-\rho)Q_d[i]$$
$$= \sum_{k=1}^{n} V_m[i]V_m[k]\big(G_{ik}\sin\big(V_a[ik]\big) - B_{ik}\cos\big(V_a[ik]\big)\big),$$

where $V_a[ik] = V_a[i] - V_a[k]$.

III. Graph Convolutional Network Model

The optimal load-shedding model in Section II takes X, $\mathbb{L}'$, and $\bar{f}$ as inputs and returns ρ as output. In this section, we describe how the data samples of form $(X, \mathbb{L}', \bar{f}, \rho)$ are preprocessed and fed into a GCN model. We also present its network architecture, as well as that of a multilayer perceptron network as a benchmark.

A. Data Preprocessing

After obtaining data samples, we preprocess them in order to make them have the right form to be trained by GCN. Specifically, we compute net real and reactive power for each bus $i \in \mathbb{N}$ as follows:

$$P_{\text{net}}[i] = \begin{cases} P_d[i] - P_g[i], & i \in \mathbb{G} \\ P_d[i], & \text{otherwise} \end{cases},$$

$$Q_{\text{net}}[i] = \begin{cases} Q_d[i] - Q_g[i], & i \in \mathbb{G} \\ Q_d[i], & \text{otherwise} \end{cases}.$$

Noting that the voltage magnitude V_m and voltage angle V_a are defined for each bus, we define a feature matrix $\hat{X} \in \mathbb{R}^{|\mathbb{N}| \times 4}$:

$$\hat{X} = (V_m, V_a, P_{\text{net}}, Q_{\text{net}}).$$

On the other hand, we construct an adjacency matrix A using the line power flow limit $\bar{f}$ and line contingency $\mathbb{L}'$ as follows:

$$A_{ij} = \begin{cases} \bar{f}[l'], & l' = (i,j) \text{ or } (j,i) \in \mathbb{L}' \\ 0, & \text{otherwise} \end{cases}.$$

After obtaining A, we apply the *renormalization trick* [6] on the adjacency matrix. The renormalization of adjacency matrix $\hat{A}$ is achieved by $\hat{A} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$, where $D_{ii} = \max_j(A_{ij})$, $\tilde{A} = A + D$, $\tilde{D}_{ii} = \Sigma_j \tilde{A}_{ij}$.

B. Network Architecture

Using $(\hat{X}, \hat{A})$ as input, we develop a GCN model that predicts ρ. Our model consists of two GCN layers followed by a fully connected network with a single hidden layer, as depicted in Figure 1. This network architecture is different from that of multilayer perceptron (MLP) in Figure 2 in that its input data $(\hat{X}, \hat{A})$ has a graph form and the underlying topology is used to generate intermediate features, whereas MLP uses just the vectorization of $\hat{X}$ and $\hat{A}$ as input and does not utilize the topology information.

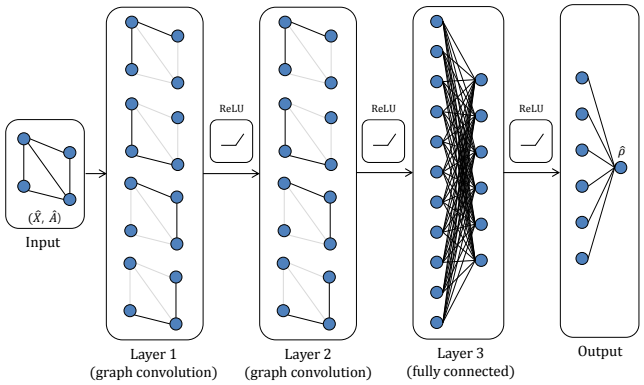

Fig. 1. Network Architecture (GCN)

As shown in Figure 1, our model takes $(\hat{X}, \hat{A})$ and outputs $Y_1 \in \mathbb{R}^{|\mathbb{N}| \times K}$ in the first layer. To generate Y_1, we first obtain a mixed feature matrix $\hat{A}\hat{X}$ that linearly combines a feature vector of each bus with those of neighboring buses using the weights provided by $\hat{A}$. Then, we extract a new set of features $\hat{A}\hat{X}W_1 + b_1$ by multiplying a weight matrix $W_1 \in \mathbb{R}^{4 \times K}$ and adding a bias vector $b_1 \in \mathbb{R}^K$. After applying the ReLU activation function $f(x) = \max(0, x)$, we obtain the first hidden unit Y_1:

$$Y_1 = \text{ReLU}(\hat{A}\hat{X}W_1 + b_1).$$

In the same way, we generate $Y_2 \in \mathbb{R}^{|\mathbb{N}| \times L}$ in the second graph convolutional layer:

$$Y_2 = \text{ReLU}(\hat{A}Y_2W_2 + b_2),$$

where $W_2 \in \mathbb{R}^{K \times L}$ and $b_2 \in \mathbb{R}^L$.

After having two GCN layers, we connect a fully connected neural network with a single hidden layer. Before entering the fully connected network, we vectorize the second hidden unit Y_2 to $\hat{Y}_2 \in \mathbb{R}^{1 \times |\mathbb{N}|L}$ and use $\hat{Y}_2$ as input to the following layer. In the third layer, we generate $Y_3 \in \mathbb{R}^{1 \times M}$ using a weight matrix $W_3 \in \mathbb{R}^{|\mathbb{N}|L \times M}$, a bias vector $b_3 \in \mathbb{R}^M$, and the ReLU activation:

$$\hat{Y}_3 = \text{ReLU}(\hat{Y}_2W_3 + b_3).$$

By multiplying a weight vector $W_4 \in \mathbb{R}^{M \times 1}$ and adding a bias $b_4 \in \mathbb{R}$, we have a network output $\hat{Y}_4 \in \mathbb{R}$:

$$\hat{Y}_4 = \hat{Y}_3W_4 + b_4,$$

which predicts an optimal load-shedding ratio ρ.

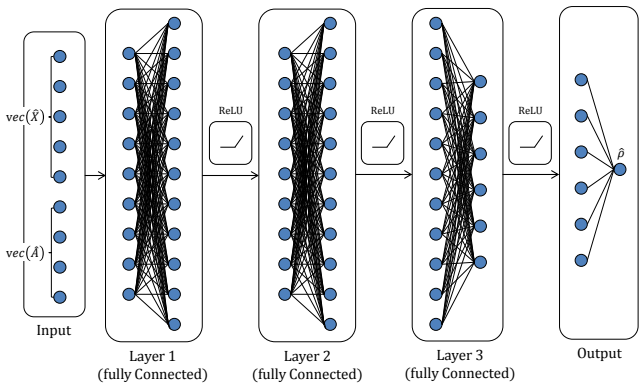

Fig. 2. Network Architecture (MLP)

IV. Numerical Experiments

We present the experimental results from benchmarking the GCN model with the MLP model and a linear regression (LR) model on the IEEE test systems with 9, 30, 57, and 118 buses.

A. Experiment Settings

The optimal load-shedding model is implemented in Julia based on the power grid examples in StructJump [7] and solved by IPOPT [8]. The GCN model is built in Python by using TensorFlow [9].

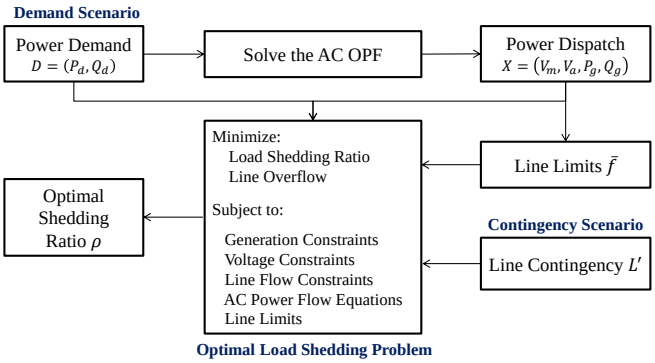

Fig. 3. Data Generation Procedure

B. Scenario Generation

We describe how to generate scenarios each of which represents $(X, \mathbb{L}', \bar{f}, \rho)$. The scenario generation steps are summarized as follows (also shown in Figure 3).

1. Using the default load profile $\bar{D} = (\bar{P}_d, \bar{Q}_d)$ in the IEEE test data, generate active and reactive power demand $D[i] = (P_d[i], Q_d[i])$ for each node $i \in \mathbb{N}$ as
$$P_d[i] = \max(\tilde{P}_d[i], 0), \quad Q_d[i] = \tilde{Q}_d[i],$$
where
$$\tilde{P}_d[i] \sim \mathcal{N}(\bar{P}_d[i], 0.01 \times |\bar{P}_d[i]|^2),$$
$$\tilde{Q}_d[i] \sim \mathcal{N}(\bar{Q}_d[i], 0.01 \times |\bar{Q}_d[i]|^2).$$
2. Solve the ACOPF problem with the load profile D, and obtain a power dispatch solution $X = (V_m, V_a, P_g, Q_g)$.
3. Compute the power flow $f[l]$ for each line $l = (f, t) \in \mathbb{L}$ as
$$f[l] = \sqrt{\frac{|S[f,t]|^2 + |S[t,f]|^2}{2}},$$
where $S[f, t]$ represents apparent power on line l from bus f to t and vice versa.
4. Set the line limit $\bar{f}[l]$ for each line $l \in \mathbb{L}$ by
$$\bar{f}[l] = (1 + \alpha[l]) f[l],$$
where $\alpha[l]$ is randomly chosen between 0 and 1.
5. Trigger a line outage for one line at a time, and define a set of active lines $\mathbb{L}' \subset \mathbb{L}$ reflecting the line contingency. This generates $|\mathbb{L}|$ contingency scenarios for the load profile D.
6. For each line contingency $\mathbb{L}'$, solve the optimal load-shedding model with the load profile D, power dispatch X, and line flow limit $\bar{f}$.

Following this procedure, we obtain $|\mathbb{L}|$ data samples at a time. To acquire a variety of data samples, we generate 100 demand scenarios for each IEEE bus system and repeat the process for each demand scenario.

The data generation results are summarized in Table I. We split the data samples into 70% and 30% for training and testing the model, respectively. Note that the number of data samples is smaller than $100 \times |\mathbb{L}|$ since the optimal load-shedding problem can be infeasible for some demand and contingency scenarios.

TABLE I
DATA GENERATION RESULTS

System	No. of lines	No. of Data Samples		
		Total	Training	Test
9-bus	9	600	420	180
30-bus	41	3900	2730	1170
57-bus	80	6889	4822	2067
118-bus	186	17683	12378	5305

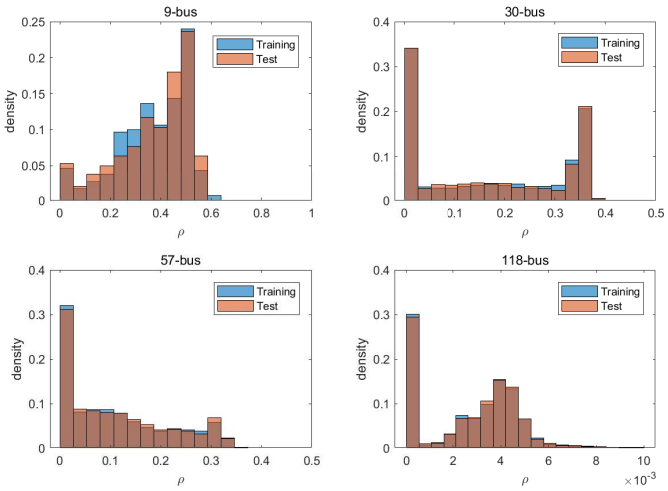

Fig. 4. Distribution of Optimal Shedding Ratio ρ

C. Data Distributions

Figure 4 displays the distributions of the optimal load-shedding ratio ρ of training and testing sets for each IEEE bus system where the mixed color represents the over-lapping area between two distributions. As seen in the figure, there is little difference between the distributions of training sets and testing sets. We note that many data samples have a near-zero value of ρ, which means that no load shedding is required for such cases. The relative portion of such data samples appears to be large for large bus systems, indicating that the large bus systems (30-bus, 57-bus, 118-bus) are more robust to a single line failure.

Regarding the distribution of the load-shedding ratio ρ excluding the near-zero values, each system has a different shape. The 9-bus system has a negatively skewed distribution whereas the 118-bus system has a bell-shaped one. The 30-bus and 57-bus systems have similar distributions, namely, a uniform distribution with a spike, but the magnitude of the spike is bigger for the 30-bus system.

D. Prediction Results

Table II and Figures 5 and 6 show the prediction results of three prediction models—GCN, MLP, and LR—on the IEEE test systems. Table II displays the root mean squared errors (RMSE) of each model for each bus system. As shown in the table, GCN outperforms MLP and LR in both training RMSE and testing RMSE for all cases. It is interesting to note that GCN not only works well on the training sets but also generalizes well to the test sets. This can be attributed to the incorporation of network topology since it guides the model to

TABLE II
PREDICTION RESULTS (RMSE)

System	Model	RMSE (Training)	RMSE (Testing)
9-bus	GCN	0.0282	0.0685
	MLP	0.0940	0.1209
	LR	0.0929	0.0980
30-bus	GCN	0.1330	0.1457
	MLP	0.1375	0.1582
	LR	0.1446	0.1734
57-bus	GCN	0.0744	0.1231
	MLP	0.1158	0.6394
	LR	0.1507	0.5251
118-bus	GCN	0.0086	0.0291
	MLP	0.1034	6.5755
	LR	0.2422	3.1741

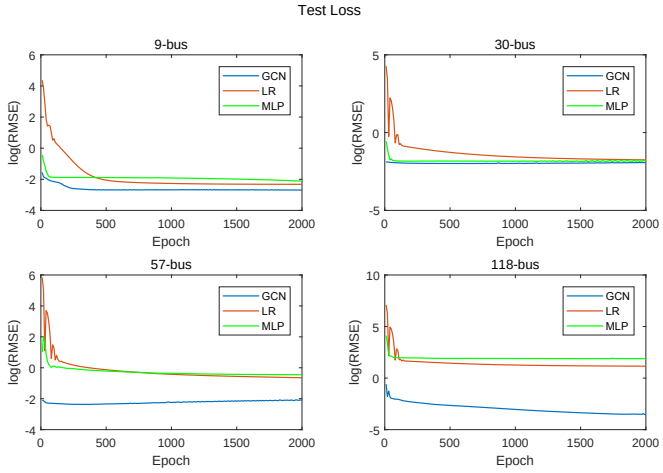

Fig. 5. Test loss of each model over the training epoch for each test instance

develop useful features to be fed to the fully connected layer. The exceptional generalization performance of GCN is well-captured in the large systems such as the 57-bus and 118-bus systems, reported in Table II. The testing RMSEs of GCN are lower by a factor of 5 and 327 than that of MLP for the 57- and 118-bus systems, respectively.

On the other hand, Figure 5 shows that the test loss of GCN rapidly decreases and stays at the near zero value. To the contrary, LR and MLP take longer to converge and they are stuck at the spurious local minimum as the number of buses increases. Figure 6 shows the distribution of test prediction errors, where the variance of the prediction errors resulting form GCN appears smaller than that of MLP and LR while centering at zero. Similar to the observations in Table II, the prediction errors are in much smaller boxes in larger systems (i.e., 57- and 118-bus systems), as compared with the boxes for the other smaller systems. These results clearly demonstrate the superior performance of GCN for predicting the load-shedding ratio ρ across the systems.

V. SUMMARY AND DIRECTIONS OF FUTURE WORK

In this work, we introduced an application of a graph convolutional network (GCN) to load-shedding operations. We presented a data generation procedure for high-quality data

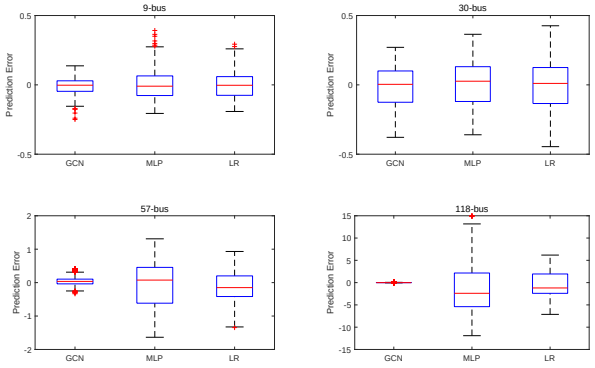

Fig. 6. Box Plots of Test Prediction Error

samples, which involves solving the optimal load-shedding problem. We described the model development process for data preprocessing, and we presented the network architecture. Experimental results on the IEEE test systems show that this model outperforms the other two benchmark models MLP and LR in predicting an optimal shedding ratio for a variety of scenarios. These results demonstrate that GCN, which takes system topology into account, can be a good choice for modeling power grid operations. This work provides a number of potential directions for future research that involve applying GCN and its variants for power system operations and analysis.

REFERENCES

[1] F. Lin, "Optimal control of load shedding in smart grids," *arXiv preprint arXiv:1712.00536*, 2017.

[2] C.-p. Lee and S. J. Wright, "Using neural networks to detect line outages from PMU data," *arXiv preprint arXiv:1710.05916*, 2017.

[3] Y. Zhao, J. Chen, and H. V. Poor, "A learning-to-infer method for real-time power grid topology identification," *arXiv preprint arXiv:1710.07818*, 2017.

[4] G. Rovatsos, X. Jiang, A. D. Domínguez-García, and V. V. Veeravalli, "Statistical power system line outage detection under transient dynamics," *IEEE Transactions on Signal Processing*, vol. 65, no. 11, pp. 2787–2797, 2017.

[5] T. Kim and S. J. Wright, "PMU placement for line outage identification via multinomial logistic regression," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 122–131, Jan 2018.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[7] J. Huchette, M. Lubin, and C. Petra, "Parallel algebraic modeling for stochastic optimization," in *Proceedings of the 1st First Workshop for High Performance Technical Computing in Dynamic Languages*. IEEE Press, 2014, pp. 29–35.

[8] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[9] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/