Learning Solutions for Intertemporal Power Systems Optimization with Recurrent Neural Networks

Mostafa Mohammadian, Kyri Baker
University of Colorado Boulder
Boulder, CO, USA
{mostafa.mohammadian, kyri.baker}@colorado.edu

My H. Dinh, Ferdinando Fioretto
Syracuse University
Syracuse, NY, USA
{mydinh, ffiorett}@syr.edu

***Abstract*—Learning mappings between system loading and optimal dispatch solutions has been a recent topic of interest in the power systems and machine learning communities. However, previous works have ignored practical power system constraints such as generator ramp limits and other intertemporal requirements. Additionally, optimal power flow runs are not performed independently of previous timesteps - in most cases, an OPF solution representing the current state of the system is heavily related to the OPF solution from previous timesteps. In this paper, we train a recurrent neural network, which embeds natural relationships between timesteps, to predict the optimal solution of convex power systems optimization problems with intertemporal constraints. In contrast to traditional forecasting methods, the computational benefits from this technique can allow operators to rapidly simulate forecasts of system operation and corresponding optimal solutions to provide a more comprehensive view of future system states.**

***Index Terms*—Recurrent neural networks, learning optimal solutions, power systems forecasting**

I. Introduction

As the capacity of installed intermittent generation in the electric power grid rises, our ability to match supply and demand on faster timescales must subsequently follow suit. Optimal power flow (OPF) problems are typically solved by grid operators to economically match supply and demand, but as fluctuations in the power supply increase, solving these problems becomes more challenging. Harnessing the benefit of historical data and moving most of the computational burden to an off-line setting, some recent work has focused on using machine learning (ML) to train models that map system loading conditions onto OPF solutions [1]–[5]. Efforts to assist, rather than replace, existing grid optimization procedures have also been developed, such as techniques to help warm-start OPF [6]–[8] to help expedite convergence.

While these works provide promising results for replacing or augmenting traditional OPF procedures with trained black-box models, they suffer from a couple flaws: First, practical power systems operations are not "snapshot" OPFs that consider a single loading scenario in isolation - intertemporal constraints, such as generator ramp limits, also can highly constrain the set of feasible system operating points [9], and future generation dispatch is highly related to the current generator dispatch. Second, ML-based tools are unfortunately a far cry away from replacing current operator practice and expertise; ML can presently, more realistically, be a supplementary tool that can help and assist operation. Thus, in this paper, we recognize the benefit of ML-based tools to expedite grid operations and turn towards 1) incorporating intertemporal constraints into these models; and 2) utilizing these models for assisting (through rapid forecasting of possible system states), rather than replacing, current optimal power flow procedures.

In contrast to previous works which focus on (deep) neural networks (DNNs) [1]–[4], here, we leverage recurrent neural networks (RNNs), which better capture temporal relationships and dependencies. RNNs were previously used in power systems applications for emulating nonlinear solvers to solve OPF problems [5] and for forecasting of demand [10], [11] and renewable energy [12]. However, these works only forecast *inputs* to the OPF problem - if an operator desired to obtain forecasts of how the system would behave (e.g., forecasting *solutions* to the OPF problem), they would have to input these forecasts into a traditional solver. Here, we combine the benefits of previous work in predicting snapshot OPF solutions with the benefits of using deep learning for forecasting future system states. In this paper, we use RNNs to forecast intertemporal (time-linked) OPF solutions, which allows operators to not only obtain estimates of resource availability and demand, but also estimates of the state of system operation.

Rather than using traditional forecasting techniques like ARIMA and ARMA, we can use ML to simultaneously approximate forecasts and optimal solutions while considering physical constraint violations during training. We show our results on the IEEE 14-bus and 57-bus networks and demonstrate the computational benefits and accuracy of using a constrained RNN-based approach for predicting the solutions of intertemporally constrained DC-OPF problems versus traditional forecasting techniques. Due to the speed of inference, this can be used to generate forecasts rapidly under a variety of possible system conditions.

II. Recurrent Neural Networks

Recurrent neural networks, also known as RNNs, are a class of neural networks (ML models) that allow previous outputs to be used as inputs while having hidden states. Fig. 1 shows the architecture of a sample RNN. For each timestep t, the hidden layer $\boldsymbol{h}_t$ and the output vector $\boldsymbol{y}_t \in \mathbb{R}^m$ are expressed

as follows:

$$\boldsymbol{h}_t = \pi_1(\boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{b}_h) \tag{1a}$$

$$\boldsymbol{y}_t = \pi_2(\boldsymbol{W}_{hy}\boldsymbol{h}_t + \boldsymbol{b}_y) \tag{1b}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the input vector, $\boldsymbol{W}_{hh}$, $\boldsymbol{W}_{xh}$, $\boldsymbol{W}_{hy}$, $\boldsymbol{b}_h$, $\boldsymbol{b}_y$ are the weight matrices and bias vectors, respectively, which are shared across time. The $\pi(\cdot)$ is the activate function which is typically non-linear (e.g., a rectified linear unit (ReLU)).

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network designed for and capable of learning order dependence in sequence prediction problems. This capability suggests that the promise of recurrent neural networks is to learn the temporal context of input sequences (past time-steps) in order to make better predictions. An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks (cells) that keeps the information about past inputs for an amount of time that is not fixed a priori. This feature differentiates them from regular multilayer neural networks, like feed-forward neural networks, that does not have memory and can only learn a mapping between input and output patterns. Although standard RNNs fail to learn in the presence of time lags greater than 5 – 10 discrete time steps between relevant input events and target signals (due to vanishing gradients or exploding gradients), LSTMs are impressive at capturing over long-term temporal dependencies without suffering from the optimization hurdles that plague the standard RNNs.

Given the rise of smart electricity meters and the broad adoption of electricity generation technology (mostly penetration of renewable sources like solar panels), there is a abundance of electricity usage data available. This data represents a multivariate time series of power-related variables, e.g. aggregated load at each node of the system or power generations at each bus and so on, that in turn could be used to model and even forecast future electricity consumption and generation. Also, the future state of the power system networks is heavily related to the previous state and timesteps due to the intertemporal constraints that link the states together. Therefore, running optimal power flow independently of previous timesteps for obtaining the current state of the system would not lead to a feasible and optimal solution. Indeed, recurrent neural networks have attained good results in a variety of applications because they can model the dynamics of the data. In the power system application, the LSTM networks can model the temporal changes in output power of generators or load profiles because of their recurrent architecture and memory units. Moreover, unlike the traditional recurrent neural networks, LSTMs were designed to avoid the long-term dependency problem. Therefore, this feature and computational benefit from this technique allow operators to rapidly simulate a wide variety of future system states dependent on how many previous ones which the system is linked to.

III. RNNs for Intertemporal DC-OPF

In this section we provide the necessary background of an intertemporal DC optimal power flow (OPF) problem with generator ramping constraints and discuss how it is made amenable for use with an RNN.

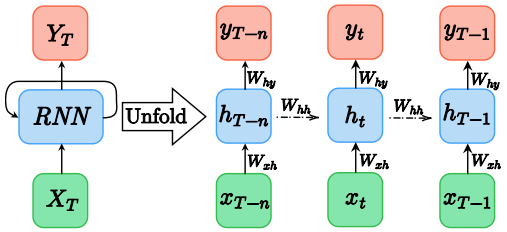


Fig. 1: A common structure of recurrent neural networks.

A. DC Optimal Power Flow

In the DC-OPF problem, we aim to determine the least-cost generator dispatch that meets the load (demand) in a power network subject to power flow limits on the network lines and power generator constraint. First, consider a power network with N buses collected in set $\mathcal{N}$, and l transmission lines collected in the set of edges $\mathcal{E}$. The horizon that it is considered for solving the problem is divided into T equal time slots. The set of time steps is denoted by $\mathcal{T} = \{1, \ldots, T\}$. The power generated and the load demand in bus $i \in \mathcal{N}$ and time $t \in \mathcal{T}$ are denoted by $p^g_{i,t}$ and $p^d_{i,t}$, respectively. UR_i and DR_i represents the ramp-up and ramp-down rate limit of unit i. $p^f_{ij,t}$ describes the active power flows associated with each transmission line. Finally, $\boldsymbol{\theta_t} \in \mathbb{R}^n$ collects the phase angles at each of i buses, with $\theta_{0,t}$ denoting the angle at the reference bus associated with each time. Therefore, the DC-OPF problem can be formulated as:

$$\underset{\boldsymbol{\theta_t}, \{\boldsymbol{p^g_t}\}_{t\in\mathcal{T}}}{\text{minimize:}} \quad \text{cost}(\boldsymbol{p^g}) = \sum_{i\in\mathcal{N}, t\in\mathcal{T}} c_{i,t}(p^g_{i,t}) \tag{2}$$

subject to

$$p^g_{i,t} - p^d_{i,t} = \sum_{(ij)\in\mathcal{E}} p^f_{ij,t} \qquad i \in \mathcal{N}, t \in \mathcal{T} \tag{3}$$

$$\underline{p}^g_i \le p^g_{i,t} \le \overline{p}^g_i \qquad i \in \mathcal{N}, t \in \mathcal{T} \tag{4}$$

$$p^g_{i,t} - p^g_{i,t-1} \le UR_i \qquad i \in \mathcal{N}, t \in \mathcal{T} \tag{5}$$

$$p^g_{i,t-1} - p^g_{i,t} \le DR_i \qquad i \in \mathcal{N}, t \in \mathcal{T} \tag{6}$$

$$p^f_{ij,t} \le \overline{p}^f_{ij} \qquad (ij) \in \mathcal{E}, t \in \mathcal{T} \tag{7}$$

$$p^f_{ij,t} = b_{ij}(\theta_{i,t} - \theta_{j,t}) \qquad (ij) \in \mathcal{E}, t \in \mathcal{T} \tag{8}$$

$$\theta_{0,t} = 0 \qquad (i) \in \mathcal{N}, t \in \mathcal{T} \tag{9}$$

where $\underline{p}^g_i$ and $\overline{p}^g_i$ represent the the minimum and maximum active power injection from the generator. The objective function (2) representing the cost of the generator dispatch, is assumed to be a quadratic function. Constraint (3) shows the power balance equation in bus $i \in \mathcal{N}$ at time step $t \in \mathcal{T}$. Constraints (4) - (9) are the operation constraints. The generators output powers, the increase or reduction in the generators output powers per time-step, and the power flowing through the transmission lines $p^f_{ij,t}$ are bounded.

B. DC-OPF Learning Goals and Modeling the Constraints

Given the inputs, the goal is to predict the current or future time-steps states of the power system. The resulting model (predictor) learns a DC-OPF mapping $\mathcal{O} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, where n is the number of load buses and generators of lag timesteps gathered as input (demand at each bus and active power injection from the generators), and m is number of the same variables in the prediction timesteps ahead horizon. The inputs given to the problem is a series of points collected in a set $\mathcal{D} = \{(\boldsymbol{x}_k, \boldsymbol{y}_k)\}_{k=1}^{K}$ where $\boldsymbol{x}_k = (\boldsymbol{p}^d_{(t-\tau_x)}, \boldsymbol{p}^g_{(t-\tau_x)} \cdots, \boldsymbol{p}^d_{(t-1)}, \boldsymbol{p}^g_{(t-1)})$ and $\boldsymbol{y}_k = (\boldsymbol{p}^d_{(t)}, \boldsymbol{p}^g_{(t)} \cdots, \boldsymbol{p}^d_{(t+\tau_y)}, \boldsymbol{p}^g_{(t+\tau_y)})$ represents the k^{th} samples in the dataset that satisfy $\boldsymbol{y}_k = \mathcal{O}(\boldsymbol{x}_k)$. The output is a function $\hat{\mathcal{O}}$ that ideally would be the result of the following optimization problem

$$\text{minimize: } \sum_{k=1}^{K} \mathcal{L}_o(\boldsymbol{y}_k, \hat{\mathcal{O}}(\boldsymbol{x}_k)) \tag{10}$$

where in the case of a recurrent neural network, the loss function $\mathcal{L}$ of all time steps is specified based on the loss at every time step as follows:

$$\mathcal{L}_o(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \sum_{\tau_y=0} ||\boldsymbol{p}^g_{(t+\tau_y)} - \hat{\boldsymbol{p}}^g_{(t+\tau_y)}|| + ||\boldsymbol{p}^d_{(t+\tau_y)} - \hat{\boldsymbol{p}}^d_{(t+\tau_y)}||$$

The *hat* notation is adopted to denote the predictions of the model, and variables without a hat denote the corresponding true values.

A baseline RNN model can be obtained by ignoring the problem (DC-OPF) constraints and minimizing the loss function. It will yield an approximation $\hat{\mathcal{O}}$ which will typically not satisfy the DC-OPF constraints, as minimizing prediction error does not necessarily ensure constraint satisfaction. Therefore, a major challenge of the learning task is to design an RNN model that incorporates hard constraints, including inequality and equality constraints within and across timesteps. To capture these constraints, this paper uses the method introduced in [3]. This technique utilizes a Lagrangian relaxation approach based on constraint violations used in a generalized augmented Lagrangian relaxation. The violation-based Lagrangian relaxation formulation is then given by

$$\text{minimize: } f(\boldsymbol{x}) + \lambda_h |h(\boldsymbol{x})| + \lambda_g \max(0, g(\boldsymbol{x}))$$

where λ_h and $\lambda_g \geq 0$ are the Lagrangian multipliers. Unlike the traditional Lagrangian relaxation which exploits the satisfiability degrees of constraints, the violation-based Lagrangian relaxation is expressed in terms of violation degrees.

The violation degree is always non-negative and indicates how much the constraint is violated. The violation degree of a constraint c can be defined as a function $\nu_c : \mathbb{R}^n \rightarrow \mathbb{R}^+$ such that $c(\boldsymbol{x})$ holds whenever $\nu_c(\boldsymbol{x}) \equiv 0$. Based on this definition, the violation degrees for inequality (denoted by the $\geq$ superscript) and equality (denoted by the $=$ superscript) constraints are determined by:

$$\nu^{=}_c(\boldsymbol{x}) = |\delta_c(\boldsymbol{x})| \tag{11}$$

$$\nu^{\geq}_c(\boldsymbol{x}) = \max(0, \delta_c(\boldsymbol{x})) \tag{12}$$

Although the resulting expressions are not differentiable everywhere, computational experiments in [3] demonstrated that violation degrees are more appropriate for predicting OPFs than satisfiability degrees. It is worth mentioning that an augmented Lagrangian method also uses both the satisfiability and violations degrees in its objective.

To define the violation degrees of the DC-OPF constraints, the baseline model in (10) needs to be developed. Given the predicted values for active power generation and load, the constraints can be captured naturally in terms of satisfiability and violation degrees. For instance, the satisfiability degree of a constraint in (3) can be obtained as:

$$\delta_3(\hat{p}^g_{i,t}, \hat{p}^d_{i,t}) = \hat{p}^g_{i,t} - \hat{p}^d_{i,t} - \sum_{(ij)\in\mathcal{E}} p^f_{ij,t}$$

for all $i \in \mathcal{N}$, the violation degree becomes

$$\nu_3(\hat{\boldsymbol{p}}^q, \hat{\boldsymbol{p}}^d) = \sum_{(i\in\mathcal{N})} \nu^{=}_c(\delta_3(\hat{p}^g_{i,t}, \hat{p}^d_{i,t}))$$

The loss function used to train DC-OPF RNN (10) can now be derived systematically, which include violation degrees of the physical and engineering constraints. Here, intertemporal generator ramp constraints and intratemporal balance constraint of the DC-OPF problem are considered in the loss function. Therefore, for the set $\mathcal{C}$ of DC-OPF constraints, the model loss is captured by the expression

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \hat{\boldsymbol{y}}) = \mathcal{L}_o(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \sum_{c\in\mathcal{C}} \lambda_c \nu_c(\boldsymbol{x}, \hat{\boldsymbol{y}}) \tag{13}$$

where $\nu_c(\boldsymbol{x}, \hat{\boldsymbol{y}})$ is the violation degree of constraint c for input $\boldsymbol{x}$ and prediction $\hat{\boldsymbol{y}}$.

IV. Simulation Results

Here, two networks were considered: The IEEE 14-bus and IEEE 57-bus. Data for these two systems were obtained via the Power Grid Library (PGLib) repository [13]. Network properties were obtained directly from this repository. Although these networks did not have the ramping capacity values to begin with, ramp limit constraints are added as these are physical engineering constraints typical of thermoelectric power generation. The dataset is split into training and testing datasets. A total of 70% of the samples of these benchmarks are used to train the model for forcasting the power generation and demand, while the remaining samples are used for testing the model. The experiments report results on the test set.

We used the root-mean-square error (RMSE) to evaluate the performance of the forecasting models. The models based on the RNN were implemented and trained using Keras package in Python 3.7 on a personal computer (Apple Laptop with M1 chip), the Adam optimizer [14] with the default learning rate $lr = 10^{-3}$ and a maximum of 250 epochs. The model architecture consist of 5 hidden layers, 3 LSTM blocks operating on inputs followed by 2 dense prediction layers with three-headed linear outputs. The LSTM layers have 128, 64, and 32 cell units and the number of nodes in the dense layers are 64 and 32, respectively. All dense layers except the linear

output layer used rectified linear unit (ReLU) activations. 12 lag timesteps of the active power generation and demand of the system have been used as input to the model for all the test cases. The forecasting is done for different two intervals (one-step and 12-steps ahead) of future generation and demand. Note that although the size of the considered networks is small in the number of buses (14 and 57 buses) for a snapshot OPF, when considering optimization across multiple timesteps, the number of required predictions (and thus size of the RNN) grows significantly. For example, in the 12-timestep ahead prediction for the 14-bus, over 200 variables are needed as inputs and as outputs to the RNN; over 1000 RNN inputs and outputs exist for the 57-bus network in this scenario.

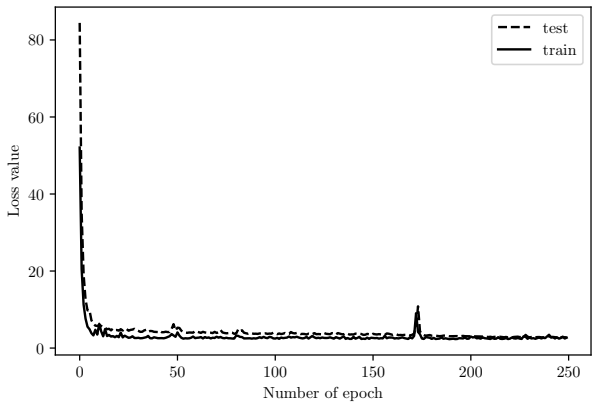
A. Dataset Generation and Training

In a physical grid, historical runs would be used as the training set for the RNN, yielding an abundance of data representing a wide variety of system states; however, for simulation and testing the model, we must generate the dataset. The load profiles given from the PGLib repository is given for one instance (snapshot) of the network. Therefore, in order to create a time series dataset, a set of normalized time series coefficients are generated based on the real-world actual demand of the CAISO data which is publicly available online [15]. Therefore, the characteristics of the resulting dataset for each network represent realistic load profiles. The trials for obtaining the coefficients took place during 29^{th} March 2021 and 12^{th} April 2021 with over 4000 snapshots of the network at a 5-minute temporal resolution.

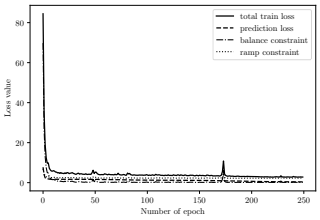
$$\boldsymbol{p}_t^d = \boldsymbol{p}_0^d \times (\text{CAISO-coefficient})_t$$

where $\boldsymbol{p}_t^d \in \mathbb{R}^{|\mathcal{N}|}$ represent the vector collecting the load demand at all load buses at time $t \in \mathcal{T}$ and $\boldsymbol{p}_0^d$ is a a base load profile. Then, due to consideration of the ramp rate constraint (5)-(6) which are intertemporal constraints, the DC-OPF detailed in (2)-(9) was run for the whole specified horizon with respect to the generated load profile. Therefore, each training point at time $t \in \mathcal{T}$ consists of a load time and its DC-OPF solution. The datasets were generated by solving DC-OPFs using the CVXPY package in Python with the Embedded Conic Solver (ECOS).

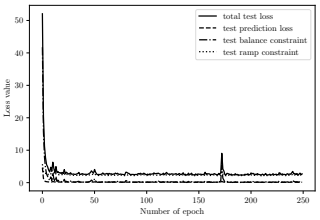
Figure 2 displays, for the IEEE 14-bus topology, the convergence of the training procedure for predicted variables including active power generations, and demand values $(\hat{\boldsymbol{p}}^q, \hat{\boldsymbol{p}}^d)$. Fig. 2a shows the total loss over the training epochs for both the train (solid line) and test (dashed line) sets. We can see that the proposed model converged reasonably quickly over a period of 50 epochs and both training and testing performance remained equivalent. The performance and convergence behavior of the model suggest that the considered loss functions (including the constraints violation) are a good match for the network learning this problem. Moreover, Figs. 2b and 2c illustrate the loss values for each of the train and test sets separately, showing how much each component of the loss function has contributed to the total loss. Using this, we can evaluate and diagnose how well the model is learning including all of the considerations of the optimization process, such as overfitting, underfitting, and convergence.



(a) Total loss



(b) Train loss



(c) Test loss

Fig. 2: Convergence analysis for train and test phases for IEEE 14-bus network in one-ahead prediction horizon.

B. Prediction Accuracy

Table I shows the root mean square error (RMSE) for the two networks using the RNN with and without constraint violations included in the loss function. As the values indicate, the inclusion of violation degrees in the loss function not only lowers constraint violations but also helps lower the overall RMSE of predictions. In addition to comparing RNNs trained with different loss functions, we also compared different traditional forecasting techniques with the RNN: a simple moving average (SMA), weighted moving average (WMA), and an AutoRegressive Integrated Moving Average (ARIMA) model (for more information on these techniques, see [16]). The ARIMA model is only capable of predicting one value per model, and thus, due to the complexity of creating over 1000 different ARIMA models for the 57-bus case, the results are only reported for the 14-bus case here. Similarly to the WMA case, the ARIMA model performs well in terms of predicting demand, but worse than the RNN with constraints in terms of predicting generation values. The RNNs also have computational benefits, which will be discussed in more detail.

In general, the RNNs outperform the traditional techniques, with a small exception of the weighted moving average for predicting demand for the 14-bus. Table II shows the prediction error for a variety of timesteps into the future with

TABLE I: RMSE (demand and active power generation) for DC OPF across the test set for one step ahead prediction.

Test Case	**Demand** (MW)	**Active Power Generation** (MW)
RNN: 14-bus with constraint inclusion	0.887	0.870
RNN: 14-bus without constraint inclusion	1.610	1.494
Simple Moving Average: 14-bus	7.787	4.372
Weighted Moving Average: 14-bus	0.871	3.822
ARIMA: 14-bus	0.780	0.901
RNN: 57-bus with constraint inclusion	3.912	3.965
RNN: 57-bus without constraint inclusion	4.211	4.719
Simple Moving Average: 57-bus	8.079	26.995
Weighted Moving Average: 57-bus	9.782	39.684

TABLE II: RMSE (demand and active power generation) for DC OPF across the entire test set for 12 step ahead predictions.

Test Case	**Timestep Ahead**	**Demand** (MW)	**Active Power Generation** (MW)
RNN: IEEE 14-bus	1	1.43	1.53
	2	1.46	1.45
	3	1.37	1.28
	4	1.29	1.3
	5	1.40	1.37
	6	1.58	1.57
	7	1.85	1.83
	8	2.12	2.19
	9	2.70	2.60
	10	3.08	3.08
	11	3.68	3.65
	12	4.12	4.17
RNN: IEEE 57-bus	1	4.06	4.11
	2	5.60	5.64
	3	7.35	7.41
	4	8.60	8.62
	5	9.60	9.64
	6	10.34	10.38
	7	10.86	10.86
	8	11.31	11.24
	9	11.61	11.53
	10	11.90	11.70
	11	11.81	11.57
	12	11.44	11.44

the proposed RNN and the WMA model. The WMA does a solid job of predicting demand for the 14-bus network, but has high power generation prediction errors. Comparing the 57-bus network, we see that the RNN captures the network complexity much better and results in significantly lower prediction errors, in particular for the generation values.

C. Constraint Violation

The objective of the RNN is to predict generation and demand values that not only have a low RMSE, but that are also physically representative. Towards this, we analyze the level of power balance and ramping constraint violation that is incurred by the predictions. Figures 3 and 4 demonstrate the ramp rates ΔP_{Gi} for $i = \{1, 2, 3\}$ in the IEEE 14-bus network. Fig. 3 shows the ramp between consecutive timesteps from generation values predicted using an RNN which considers constraint violations. Fig. 4 shows the ramp between consecutive timesteps from generation values predicted using an RNN which only considers mean squared errors. Comparing the figures, it is clear that including constraint violations during training significantly impacts the trained RNN's ability to predict generation values which satisfy these constraints. For the case with the RNN trained using constraint violation terms, Fig. 5 shows the percent error in the supply demand

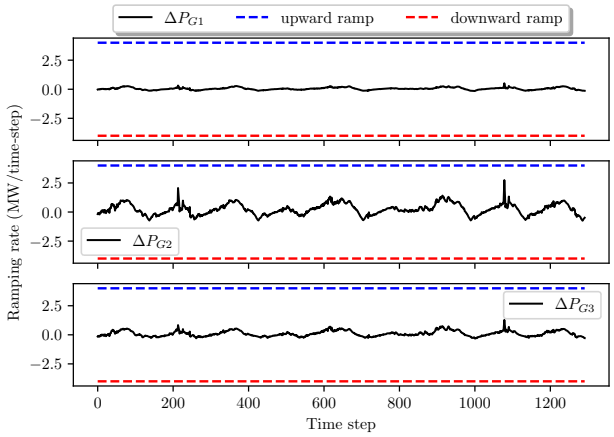


Fig. 3: Resulting ramp rates from predicted generation values for each of the generators in the IEEE 14-bus when constraint violations are included during training.

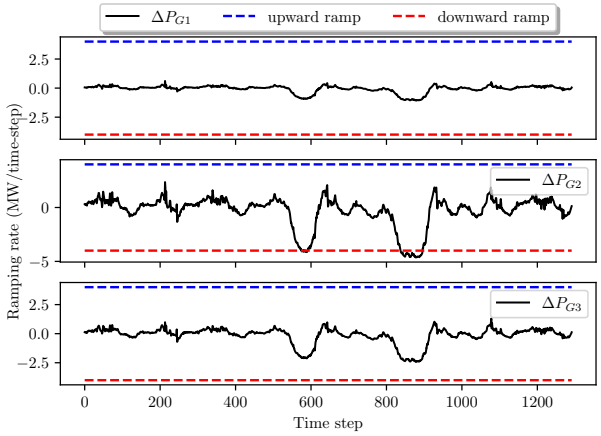


Fig. 4: Resulting ramp rates from predicted generation values for each of the generators in the IEEE 14-bus when constraint violations are not included during training.

TABLE III: Constraint violation results from the RNN with and without including constraint violations during training.

Test Case	**With Constraints**	**Without Constraints**
14-bus power balance (% mismatch)	0.3110%	0.3789%
14-bus ramp violations (of instances)	0	71
57-bus power balance (% mismatch)	0.9079%	1.5782%
57-bus ramp violations (of instances)	0	1

balance in the network across all testing samples. Generally, the mismatch is far under $\pm 1\%$. Table III tabulates these values in numerical form, with the average % mismatch between generation and demand in each considered case and the number of instances throughout the test set in which the ramp limits were violated.

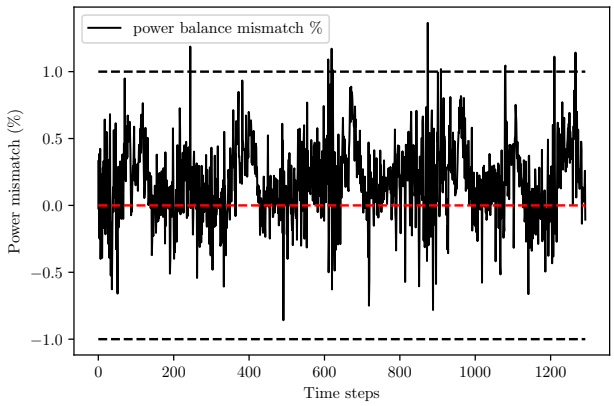


Fig. 5: Power balance mismatch from predicted generation values for the IEEE 14-bus network.

TABLE IV: Training and inference (mean and $\pm$ one standard deviation) computational costs.

Test Case	**Train time** (min)	**Predict time** (sec)
IEEE 14-bus (one-step ahead prediction)	5.229	0.241 ± 0.00552
IEEE 14-bus (12-step ahead prediction)	5.247	0.247 ± 0.00368
IEEE 57-bus (one-step ahead prediction)	6.072	0.296 ± 0.00791
IEEE 57-bus (12-step ahead prediction)	6.664	0.303 ± 0.00191
ARIMA: 14-bus	9.083	0.951

D. Computation Time

Finally, Table IV reports the average computation times for the model to be trained from the RNN and the ARIMA models. Note that training can be done offline, but the short training time indicates that the model could be re-trained to reflect more recent system conditions if necessary. Additionally, the average time needed to produce a prediction across the entire test dataset (not just a given instant) is shown in the table plus or minus one standard deviation as indicated in the case of the RNN. It is seen that there is no significant change in the training phase of one-step and multi-step ahead prediction of each test case. However, the RNN offers over three times the speed of the ARIMA model, which is the most sophisticated traditional model considered. Considering the ARIMA, SMA, and WMA methods do not consider constraint satisfaction, a grid operator may have to run a DC-OPF in order to predict actual generator behavior. These predictions, used in an online mode, are much faster than solving DC-OPFs that deal with only one snapshot of the network state. Therefore, we have to bear in mind that our proposed model provides an appealing trade-off between accuracy and efficiency.

V. Conclusion and Future Work

In this paper, we presented a framework for predicting the solution of DC optimal power flow problems with intertemporal constraints. Unlike traditional forecasting methods, the proposed RNN-based method includes physical network constraints and is capable of simultaneously predicting an entire DC-OPF solution in one pass through the model. The framework can allow for grid operators to rapidly produce forecasts of *optimal* solutions rather than forecasts of problem inputs such as renewable availability and demand. The results shown for the 14-bus and 57-bus IEEE test networks show effective tradeoffs between computational speed, accuracy, and constraint satisfaction - proving promising for future developments on larger test networks.

Future work includes extending the formulation to AC OPF problems and including different types of intertemporal constraints such as constraints on energy storage and dynamic line ratings. Another interesting direction of future work could be to quantify the impact of input uncertainty (e.g. from existing forecasting models for renewable availability and demand) on the forecasted optimal generation values and voltages.

Acknowledgements

This work was partially supported by the National Science Foundation CAREER awards 2041835 and 2143706 and NSF grant 2007164.

References

[1] P. L. Donti, D. Rolnick, and J. Z. Kolter, "DC3: A learning method for optimization with hard constraints," in *International Conference on Learning Representations (ICLR)*, 2021.

[2] A. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow," in *IEEE SmartGridComm*, Dec. 2020.

[3] M. Chatzos, F. Fioretto, T. W. K. Mak, and P. V. Hentenryck, "High-fidelity machine learning approximations of large-scale optimal power flow," *arXiv preprint arXiv:2006.16356*, 2020.

[4] F. Fioretto, T. W. Mak, and P. Van Hentenryck, "Predicting ACOPFs: Combining deep learning and lagrangian dual methods," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[5] K. Baker, "Emulating AC OPF solvers for obtaining sub-second feasible, near-optimal solutions," *arXiv preprint arXiv:2012.10031*, 2020.

[6] D. Biagioni, P. Graf, X. Zhang, A. S. Zamzam, K. Baker, and J. King, "Learning-Accelerated ADMM for Distributed DC Optimal Power Flow," *IEEE Control Systems Letters*, vol. 6, pp. 1–6, 2022.

[7] F. Diehl, "Warm-starting AC optimal power flow with graph neural networks," in *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, Dec. 2019.

[8] K. Baker, "Learning warm-start points for AC optimal power flow," in *IEEE Machine Learning for Signal Proc. Conf. (MLSP)*, Oct. 2019.

[9] S. Frank and S. Rebennack, "An introduction to optimal power flow: Theory, formulation, and examples," *IIE Transactions*, vol. 48, no. 12, pp. 1172–1197, 2016.

[10] S. Muzaffar and A. Afshari, "Short-term load forecasts using LSTM networks," *Energy Procedia*, vol. 158, pp. 2922–2927, 2019.

[11] Z. Chen *et al.*, "Load forecasting based on LSTM neural network and applicable to loads of "replacement of coal with electricity"," *Journal of Electrical Engineering & Technology*, no. 5, pp. 2333–2342.

[12] H. K. Ahn and N. Park, "Deep RNN-based photovoltaic power short-term forecast using power IoT sensors," *Energies*, vol. 14, no. 2, 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/2/436

[13] S. Babaeinejadsarookolaee *et al.*, "The power grid library for benchmarking AC optimal power flow algorithms," Aug. 2019, [Online] Available: https://arxiv.org/abs/1908.02788.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2015, [Online] Available at: https://arxiv.org/pdf/1907.02206.

[15] "California ISO: Today's outlook," last accessed 8/2021. [Online]. Available: http://www.caiso.com/TodaysOutlook/Pages/default.aspx

[16] M. E. El-Hawary, *Short-term forecasting of electricity prices using mixed models*. Wiley-IEEE Press, 2017.