Spatio-Temporal Network Forecasting

Atharva Joshi
Electrical and Computer Engineering
Northeastern University
Boston, MA 02115
joshi.at@northeastern.edu

Jason Serpe
Electrical and Computer Engineering
Northeastern University
Boston, MA 02115
serpe.ja@northeastern.edu

Kedar Ghule
Electrical and Computer Engineering
Northeastern University
Boston, MA 02115
ghule.k@northeastern.edu

Abstract

Spatio-temporal network forecasting is an interdisciplinary research area that aims to predict the behavior of complex networked systems over time and space. We use traffic forecasting as a case study for this problem. Accurate traffic forecasting helps reduce congestion, optimize travel time, and improve safety on roads, highways, and public transportation. However, accurate traffic forecasting is a complex and challenging problem due to the high variability and unpredictability of traffic flows. Previous research has shown recurrent models like Long Short-Term Memory (LSTMs) [1] being applied on these problems. However, they fail to capture the complex spatial and temporal dependencies in traffic data. We study the state-of-the-art approaches like DCRNN [2], STGCN [3] and A3TGCN [15], and propose a novel method built on top of the STGCN model. We propose using Batch Normalization [5] instead of Layer Normalization to learn complex temporal relationships between the input features better. We also propose an architectural change in the current STGCN model by replacing the single temporal convolution layer in the output layer with a Temporal Gated-Convolution layer. This will enable the model to selectively attend to different temporal features based on their relevance to the final output. An additional advantage of this is that the gating mechanisms can help preserve gradient flow and prevent it from vanishing, leading to more stable training. We use 12-step mean absolute error (MAE) as the evaluation metric for our experiments.

1 Introduction

Spatio-temporal network forecasting is a technique that combines spatial and temporal data to predict future values in a complex network system. Applications of spatio-temporal network forecasting include traffic forecasting, weather forecasting, epidemiology, and urban planning. In traffic forecasting, spatio-temporal network forecasting is used to predict traffic flow patterns and congestion levels in real-time. This information can then be used to optimize routing, reduce travel time, and improve road safety. It is accomplished by analyzing data from a network of sensors distributed over a geographic region and extracting patterns of movement and correlation between the sensors over time. The resulting model can be used to predict future values of the network, like, speed of the traffic, given input data.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

Recent research in this domain includes the use of LSTM [1], DCRNN [2], STGCN [3], and A3TGCN [15]. We implemented our baseline model as the LSTM and studied and implemented the above models to take three different research approaches - (1) Convolution and Recurrent Neural networks, (2) Spatio-Temporal Graph Convolution Networks, and (3) Graph Attention and Convolutional models. These studies helped us determine potential room for improvement in the STGCN model when applied to the METR-LA dataset. Our proposed solution builds on top of the STGCN model. We conducted exhaustive experiments of our solution on the METR-LA dataset and were able to decrease the 12-step mean absolute error by 6.49% on the original STGCN model and 48.56% on our baseline model of LSTM. The results have been presented in further sections.

In summary, the main contributions of our proposed solution include:

1. Replace Layer Normalization with Batch Normalization: We propose to replace the layer normalization in the spatio-temporal convolution block to batch normalization. Ioffe, S., & Szegedy, C [5] and Wang et al. [11] highlight the benefits of using batch normalization for convolutional neural networks.
2. Replacing the temporal convolution layer with a temporal gated convolution layer in the output layer: Yu et al. [3] introduce this layer in their paper but use it only in the ST-Conv block. We propose to use this layer in the output layer replacing the simple temporal convolution layer as a gated layer will help modulate the contribution of each temporal feature to the final output, based on its relevance to the prediction.

An open-source reference was referred to for the STGCN implementation[1] and the PyTorch Geometric Temporal [14] library was used for the the implementation of DCRNN, A3TGCN and the Temporal Gated Convolution layer.

2 Related Work

LSTM: Zhao et al. [1] first presented that LSTM-based approach could be superior to traditional traffic forecasting models and existing models that used typical recurrent neural networks (RNNs). The authors highlighted the benefits of LSTMs for analyzing and learning patterns in larger time ranges by conducting their experiments on traffic data from a busy intersection in Singapore and got significantly better results than traditional RNN models.

CNN-LSTM: Zhao et al. [6] developed a model that first passed input data through convolutional-pooling layers. These layers extracted additional spatial information and passed results to the LSTM layer. In their experiments, the authors used traffic data from a busy intersection in Beijing, China, and showed that the proposed CNN-LSTM-based model achieved a 1-2% error reduction compared to existing methods, including traditional LSTMs.

Diffusion Convolutional Recurrent Neural Network: To handle learning spatial relationships, Li et al. [2] utilized diffusion convolutions, and a sequence to sequence encoder-decoder structure to capture temporal features. Tested on the METR-LA dataset, the Diffusion Convolutional Recurrent Neural Network (DCRNN), achieved a 12-15% error reduction over traditional and current deep learning methods. The authors also achieved a 5% error reduction compared to identical models that used Graph Convolutional layers as opposed to Diffusion layers.

Graph Neural Networks: Several state-of-the-art traffic prediction models use graph neural networks. The survey paper by Wu et al. [7] reviews the modern deep learning techniques for graph data, with specific focus on graph neural networks (GNNs) and their applications. The authors discuss open-source implementations like PyTorch Geometric [9] and deep graph library (DGL) [10] which provide implementations of various state-of-the art GNNs. The review paper utilizes the METR-LA dataset [8] as one of the benchmark datasets for traffic forecasting.

Spatio-Temporal Graph Convolution Networks: Yu et al. [3] proposes a novel approach using graph convolutions and gated graph convolutions which are useful in extracting the most useful spatial and temporal features. The authors leverage spatio-temporal graph convolution layers and temporal gated layers to achieve this. The authors showed that the STGCN model outperformed several state-of-the-art methods on the PeMSD7 dataset in traffic forecasting, highlighting the potential of graph convolutional networks and gated graph convolutions for spatio-temporal learning tasks.

[1] https://github.com/FelixOpolka/STGCN-PyTorch.

Attention Temporal Graph Convolutional Network: Bai et al. [15] proposed a graph convolutional network to capture the spatial correlations among different road segments and an attention mechanism to capture the temporal correlations between adjacent time steps named Attention Temporal Graph Convolutional Network (A3TGCN). In addition, it uses GRUs to learn short-term time trends in the data. The authors highlight that attention can be used to determine the pairwise correlation between different timesteps. The authors achieved state-of-the-art performance on the SZ_taxi and Los_loop datasets.

Lastly, **Graph Multi-Attention Network (GMAN)** for Traffic Prediction by Zheng et al. [4] proposes overcoming some of the difficulties faced by other approaches like static adjacency matrix representations and error propagation through multiple steps. The goal of this line of work is to explore the use of attention to better model dynamic network spatiotemporal dependencies in the context of the traffic prediction problem. However, this model requires a user input of temporal and spatial embeddings.

3 Methods

3.1 Models Developed

Post the literature survey, we explored three different research approaches to the problem - (1) Diffusion Convolutional Recurrent Neural Network [2], (2) Spatio-temporal Graph Convolutional Network [3], and (3) Attention Temporal Graph Convolutional Network [15]. The results for each of them are summarized in the Experiments section.

3.1.1 Diffusion Convolutional Recurrent Neural Network

The first approach we explored was the Diffusion Convolutional Recurrent Neural Network (DCRNN) [2]. This model is an encoder/decoder architecture using Gated Recurrent Unites (GRU's), where the typical matrix multiplication operations are replaced with a diffusion convolution [16]. This diffusion process involves a random walk of the input graph (representing the road structure) to learn spatial relationships. The encoder/decoder structure allows the model to learn the temporal features. On the METR-LA dataset, the authors were able to achieve a 3.60 Mean Absolute Error (MAE) for a 1-hour forecast horizon. The architecture of the model is shown in Figure 1.

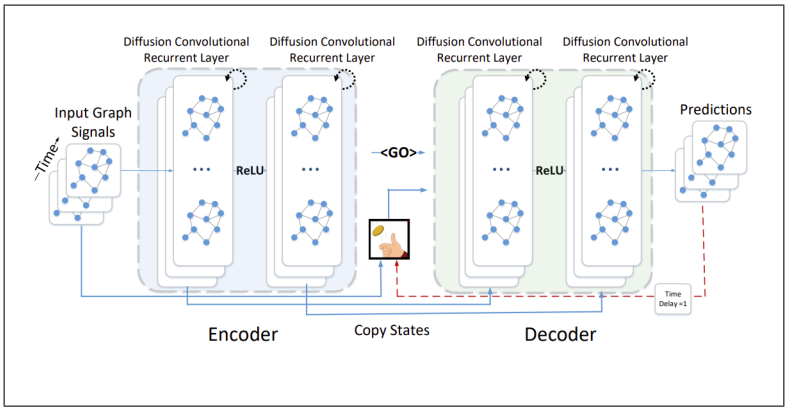

Figure 1: Architecture of DCRNN [2].

3.1.2 Spatio-Temporal Graph Convolution Network

The Spatio-Temporal Graph Convolution Network (STGCN) [3] was another approach that we implemented to compare its performance to the baseline. It was the first time a purely convolutional layer was applied to extract spatio-temporal features simultaneously from graph-structured time series for a traffic prediction problem, as claimed by the authors. The model uses two spatio-temporal convolution blocks and an output layer consisting of a temporal convolution layer and a fully connected layer. Each spatio-temporal convolution block consists of a graph convolution layer

sandwiched between two temporal gated-convolution layers followed by layer normalization. The authors did not carry out their experiments of the STGCN model on the METR-LA dataset. The architecture of this model is shown in Figure 2.

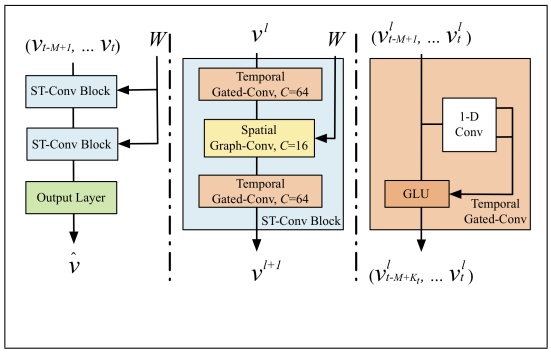

Figure 2: Architecture of STGCN [3].

3.1.3 Attention Temporal Graph Convolutional Network

The A3TGCN model [15] uses a temporal graph convolution (T-GCN) model [17] which combines Graph Convolution layers with a GRU. The hidden states are then passed through the attention layer where the context vector that computes traffic variation is determined. The weight of each spatio-temporal characteristic 'h' is calculated by softmax and a multilayer perceptron. Finally, the forecasts are outputted using a fully connected layer. The architecture of this model is shown in Figure 3.

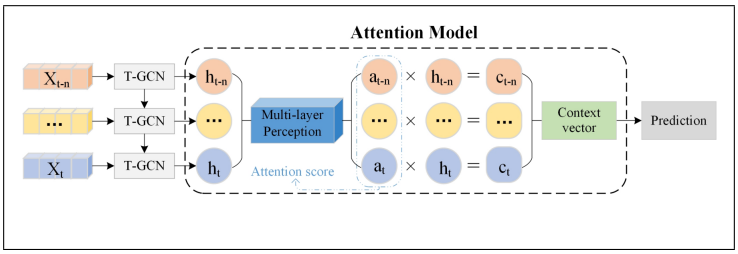

Figure 3: Architecture of A3TGCN [15].

3.2 Proposed Work

3.2.1 Network Architecture

In this section, we elaborate on the proposed modification in the architecture of spatio-temporal graph convolutional networks (STGCN) [3]. Figure 2 shows the architecture of the STGCN model proposed by Yu et al. We propose a modification to the above architecture as shown in Figure 4. We replace the layer normalization at the end of each of the spatio-temporal convolutional blocks with Batch Normalization. To capture temporal dependencies well, we replace the temporal convolution layer in the output layer with temporal gated convolution layer.

3.2.2 Batch Normalization

Batch normalization is a technique introduced by Ioffe, S., & Szegedy, C [5] that makes neural network training faster and stable. The authors highlight that this technique makes normalization a part of the model architecture and is useful for models training mini-batches of data. They demonstrate the effectiveness of batch normalization in improving the training of deep CNNs by stabilizing the gradients and improving the generalization performance of CNNs, particularly in deep architectures. Wang et al. [11] also highlight the benefits of using batch normalization for convolution neural networks and graph convolutional neural networks. They demonstrated that

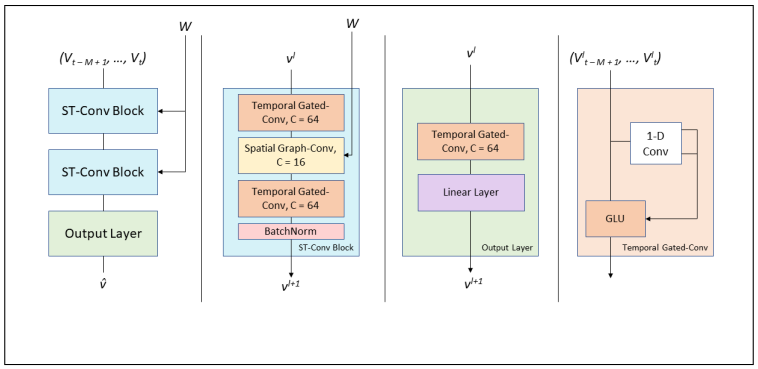

Figure 4: Proposed Architecture.

using batch normalization to the internal layer's inputs over every mini-batch is essential in order to guarantee the batch normalized output have a uniform distribution. Finally Chen et al. [13] demonstrate the importance of batch normalization to attain state-of-the-art results for traing deep GNNs on large datasets. All of these studies highlight the benefits of batch normalization for models that use mini-batch training. The reason for this is that batch normalization normalizes the inputs to the activation function using the statistics of the entire mini-batch, which helps to reduce the so-called "internal covariate shift problem" that can occur during training. Furthermore, it also adds regularization to the model. We use these studies as our intuition to replace Layer Normalization in the original STGCN model with Batch Normalization since we perform mini-batch training and our model deals with spatio-temporal relationships between the input features.

3.2.3 Temporal Gated Convolution Layer in the Output Layer

The temporal gated convolution layer introduced by Yu et al. [3] with the STGCN model is a temporal convolutional layer which contains a 1-D convolution layer with a kernel width K followed by gated linear units (GLU) as non-linearity. The authors highlight the contribution of the non-linearity gates which helps the model exploit the entire input in the stacked temporal layers. This layer also incorporates residual connections to help with any vanishing gradient problem. We studied the benefits of this layer and propose replacing the temporal convolution layer in the output with this temporal gated convolution layer. Our intuition is that having this layer before the fully connected layer can help the model selectively attend to different temporal features based on their learned relevance to the final output. The gated linear unit should help modulate the contribution of each temporal feature to the final output, based on its learned relevance to the prediction. This would allow the model to focus on the most relevant temporal features and ignore those that are less important, leading to better performance.

4 Experiments

4.1 Dataset Description

The dataset that we will use in our experiments is the METR-LA dataset [8], a popular benchmark dataset for many traffic forecasting models. This dataset is based on the Los Angeles Metropolitan traffic data and has traffic readings collected from 207 loop detectors on highways in aggregated 5-minute intervals. Each timestep is this 5 minute interval. The data was collected from March 2012 to June 2012 in the Los Angeles County, California.

4.2 Evaluation Metric

In traffic forecasting, the Mean Absolute Error (MAE) is a commonly used evaluation metric. However, it has been observed that traditional MAE does not fully capture the temporal dependencies of traffic data. To address this issue, a modified MAE called the 12-step MAE has been used as an evaluation metric in recent research. We will be using 12-step MAE to evaluate our experiments.

In this metric, the forecasted values are compared to the actual values at 12 consecutive time steps, instead of just comparing one predicted value to one actual value.

4.3 Data Preprocessing

The data was normalized using z-score normalization. This involved calculating the mean and standard deviation of the speed and time values. The values for each node at each time step were normalized.

4.4 Baseline and Results: LSTM

For our baseline model, we chose to implement an LSTM. While an LSTM is capable of learning temporal features in the data, it does not have the capacity to learn spatial features. We trained our LSTM for 50 epochs with a learning rate of 1e-6. We used the L1 loss (MAE) and the Adam optimizer. The number of LSTM layers are 3 the hidden dimension was 512. The results we achieved are summarized in Figure 5.

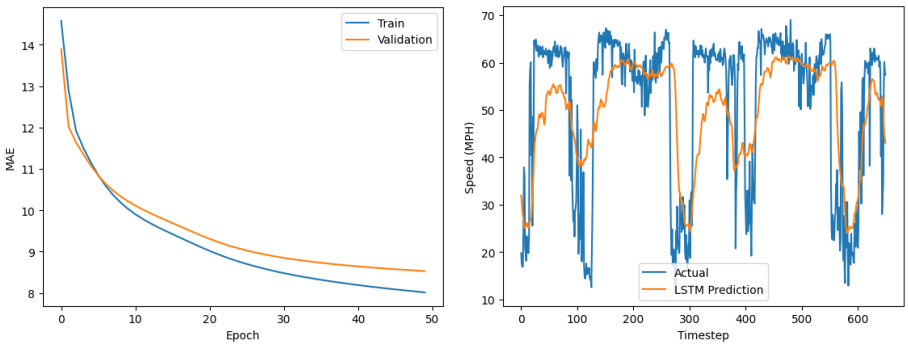

Figure 5: LSTM Learning Curve (left) and Predictions for sensor 100 (right)

As shown, the LSTM is able to roughly match the traffic pattern, but there are a couple of issues. First, it tends to 'miss' the start and end of peak hours, when the traffic has an abrupt change. Additionally, the LSTM tends to predict a narrower range of speeds. We achieved a test 12-step MAE of 9.609.

4.5 Experiments With State-of-the-Art Models and Modifications

As mentioned in the earlier sections, we explored three different approaches for the problem at hand. Their results are summarized in this sub-section.

4.5.1 DCRNN

We developed a simplified version of the DCRNN architecture proposed by Li et al. [2] using open-source implementation of the Diffusion Convolutional Gated Recurrent layer from the Pytorch Geometric Temporal [14] library. We trained our model for 50 epochs minimizing the L1 loss at each epoch using the Adam optimizer. The learning rate was 1e-2 with a decay of 0.1 at the 20th, 30th and 40th epoch using MultiStep scheduler. The convolution kernel size was 3 and the hidden dimension were 64. These are nearly the exact parameters determined optimal in the original paper [2]. However, we train for fewer epochs (50 versus 100) and thus have a different number of steps in the learning rate. The results we achieved are summarized in Figure 6.

The DCRNN variant achieved a test MAE of 5.0973 for a one-hour forecast. While this is lower than the accuracy of the model outlined in the paper (3.60), this is expected given the reduced complexity of the model. Still, it performed much better than the LSTM, demonstrating the necessity for learning spatial relationships for accurate predictions. As evidenced in the above, the Simplified-DCRNN predicts a more similar range of values and more accurately reacts to large and abrupt changes in the traffic flow.

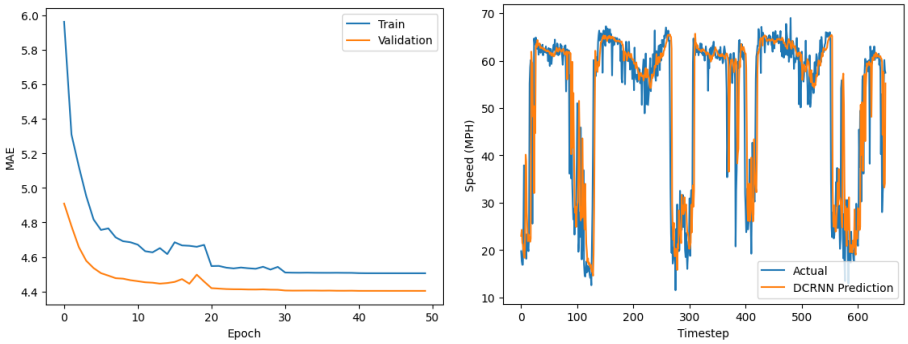

Figure 6: Simplified-DCRNN Learning Curve (left) and Predictions for sensor 100 (right)

4.5.2 STGCN

We developed the STGCN model proposed by Yu et al. [3]. Since the authors did not run the model on the METR-LA dataset and there is no documentation about the configurations of the model for the METR-LA dataset, we performed some hyperparameter tuning. The model was trained for 50 epochs with a batch size of 64. Like the original paper, we used mean squared error (MSE) as our loss function and had a learning rate of 1e-3 with a decay rate of 0.7 after every 5 epochs. We used the RMSprop optimizer. The temporal convolution kernel size and the graph convolution kernel size was kept as 3.

We achieved results shown in Figure 7. It outperforms our baseline LSTM model by achieving a 12-step MAE of 5.285. Compared to the LSTM it does a good job at predicting abrupt traffic changes.

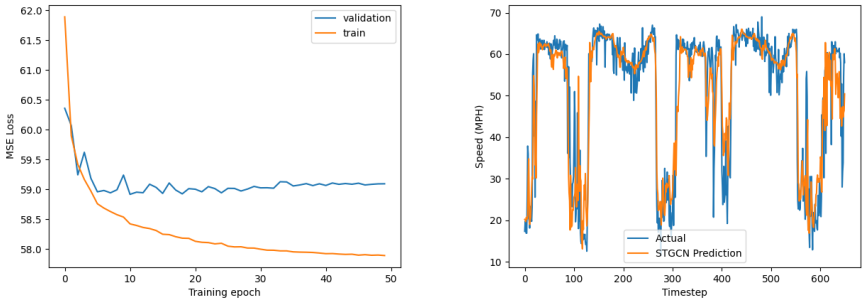

Figure 7: STGCN MSE Loss Curve (left) and Predictions for sensor 100 (right).

4.5.3 A3TGCN

We developed the A3TGCN model proposed by Bai et al. [15]. We observed that the model is unable to handle large amounts of training data even with a A100 GPU (40GB RAM) on the Discovery cluster. The high memory requirement of the model is evident in the paper where the authors train the model on a very small 7 day subset of the METR-LA dataset (Los_loop dataset). We trained our model on a 35 day subset of the METR-LA dataset for 50 epochs using a learning rate of 1e-2. The high learning rate is to maximize parameter update in minimal epochs. We minimize the MSE loss with the Adam optimizer. The results are shown in Figure 8.

We achieved a 12-step MAE of 7.841 on the 35 day subset of the data. We also experimented with the GMAN model and found it uniquely sensitive to temporal embeddings required as input. Hence GMAN was not pursued further due to potential lack of generalizability.

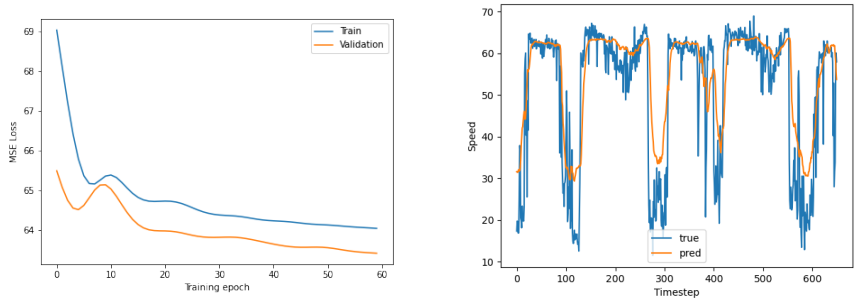

Figure 8: A3TGCN's MSE Loss Curve (left), and predictions for sensor 100 (right).

4.6 Experiment Results

Building on top of the STGCN [3] model, we propose some changes to the architecture and normalization methods. The results are summarized in this section.

4.6.1 Effect of Batch Normalization

As explained in section 3.2.2, we replace the layer normalization at the end of each ST-Conv block of the STGCN with batch normalization.

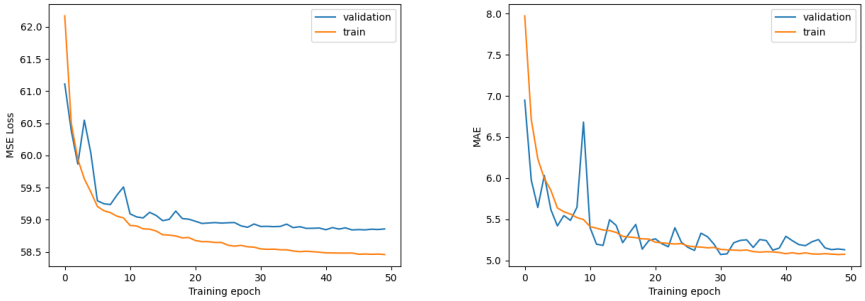

Figure 9: STGCN with BatchNorm MSE Loss Curve (left) and MAE curve (right).

Figure 9 shows our results for this experiment. We can clearly see by comparing with Figure 7 that using batch normalization helped and also introduced regularization in our model. Seeing the MSE loss curve and the MAE curve, we can see that our experiment was successful and our model could learn the spatio-temporal features better. We get a 12-step validation MAE of 5.130 which outperforms the original STGCN model.

4.6.2 Effect of Temporal Gated Convolution Layer

As explained in section 3.2.3, we next replaced the temporal convolution layer in the output layer with a temporal gated convolution layer. We used the open-source implementation of this layer available in the PyTorch Geometric Temporal [14] library for our solution.

Figure 10 shows our results for this experiment. We can see that this model can predict sudden changes in speed more precisely than the original STGCN model. We achieved a 12-step MAE of 4.942. Our proposed solution gives a 6.49% decrease in the 12-step MAE over the original STGCN model and a 48.56% decrease in the 12-step MAE over our baseline LSTM model. We use a learning rate of 0.001 with a decay rate of 0.7 over 5 epochs using StepLR. The MSE loss is minimized using the RMSprop optimizer. We also used the Adam optimizer but the results observed were to be similar.

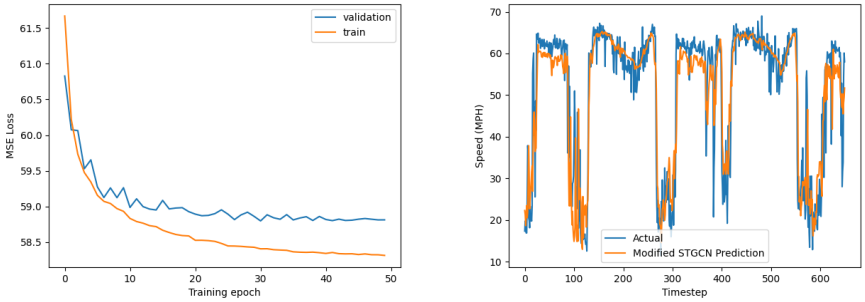

Figure 10: Proposed modified STGCN model's MSE Loss Curve (left), and predictions for sensor 100 (right).

5 Conclusion and Future Work

In this project, we studied different research approaches for the traffic prediction problem and implemented three state-of-the-art models. We propose a modification on the state-of-the-art STGCN model for traffic prediction by leveraging the power of batch normalization for convolutional and graph convolutional networks and adding an extra gated temporal convolution in the output layer. Our experiments show that we outperform our baseline model and the STGCN model. In the future, we will test our proposed solution with other benchmark traffic prediction datasets and see how the the benefits of other models like the DCRNN [2], A3TGCN [15] and GMAN [4] can be incorporated in this proposed solution. Consequently, we intend to publish our work at a reputed conference/journal.

References

[1] Zhao, Z., Chen, W., Wu, X., Chen, P. C., & Liu, J. (2017). LSTM network: a deep learning approach for short-term traffic forecast. IET Intelligent Transport Systems, 11(2), 68-75.

[2] Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2018). Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In International Conference on Learning Representations.

[3] Yu, B., Yin, H., & Zhu, Z. (2018, July). Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In Proceedings of the 27th International Joint Conference on Artificial Intelligence (pp. 3634-3640).

[4] Zheng, C., Fan, X., Wang, C., & Qi, J. (2020, April). Gman: A graph multi-attention network for traffic prediction. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 01, pp. 1234-1241).

[5] Ioffe, S., & Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning (pp. 448-456).

[6] Zhao, Z., Li, Z., Li, F., & Liu, Y. (2021, September). CNN-LSTM Based Traffic Prediction Using Spatial-temporal Features. In Journal of Physics: Conference Series (Vol. 2037, No. 1, p. 012065). IOP Publishing.

[7] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C.,& Philip, S. Y. (2020). A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems, 32(1), 4-24.

[8] Jagadish, H. V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R., & Shahabi, C. (2014). Big data and its technical challenges. Communications of the ACM, 57(7), 86-94.

[9] Fey, M., & Lenssen, J. E. (2019). Fast graph representation learning with PyTorch Geometric. arXiv preprint arXiv:1903.02428.

[10] Wang, M. Y. (2019, January). Deep graph library: Towards efficient and scalable deep learning on graphs. In ICLR workshop on representation learning on graphs and manifolds.

[11] Wang, S. H., Govindaraj, V. V., Górriz, J. M., Zhang, X., & Zhang, Y. D. (2021). Covid-19 classification by FGCNet with deep feature fusion from graph convolutional network and convolutional neural network. Information Fusion, 67, 208-229.

[12] Baldi, P., & Sadowski, P. J. (2013). Understanding dropout. Advances in neural information processing systems, 26.

[13] Chen, T., Zhou, K., Duan, K., Zheng, W., Wang, P., Hu, X., & Wang, Z. (2022). Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[14] Rozemberczki, B., Scherer, P., He, Y., Panagopoulos, G., Riedel, A., Astefanoaei, M., ... & Sarkar, R. (2021, October). Pytorch geometric temporal: Spatiotemporal signal processing with neural machine learning models. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management (pp. 4564-4573).

[15] Bai, J., Zhu, J., Song, Y., Zhao, L., Hou, Z., Du, R., & Li, H. (2021). A3t-gcn: Attention temporal graph convolutional network for traffic forecasting. ISPRS International Journal of Geo-Information, 10(7), 485.

[16] Atwood, J., & Towsley, D. (2016). Diffusion-convolutional neural networks. Advances in neural information processing systems, 29.

[17] Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., ... & Li, H. (2019). T-gcn: A temporal graph convolutional network for traffic prediction. IEEE transactions on intelligent transportation systems, 21(9), 3848-3858.