Deep Learning for Traffic Prediction with an Application to Traffic Lights Optimization

Walter Gamarra
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
gamarra.wgdg@gmail.com

Elvia Martínez
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
martinez.epmm@gmail.com

Kevin Cikel
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
kcikel@ing.una.py

Maira Santacruz
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
msantacruz@ing.una.py

Mario Arzamendia
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
marzamendia@ing.una.py

Derlis Gregor
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
dgregor@ing.una.py

Marcos Villagra
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
mdvillagra@fiuna.edu.py

José Colbes
Faculty of Engineering
National University of Asuncion
San Lorenzo, Paraguay
jcolbes@fiuna.edu.py

***Abstract*—This work proposes the use of deep neural networks for the prediction of traffic variables for measuring traffic congestion. Deep neural networks are used in this work in order to determine how much time each vehicle spends in traffic, considering a certain amount of vehicles in the traffic network and traffic light configurations. A genetic algorithm is also implemented that finds an optimal traffic light configuration. With the implementation of a deep neural network for the simulation of traffic instead of using a simulation software, the computation time of the fitness function in the genetic algorithm improved considerably, with a decrease of precision of less than 10%. Genetic algorithms are used in order to show how useful deep neural networks models can be when dealing with vehicular flow slowdown.**

***Index Terms*—traffic simulation, deep learning, genetic algorithms**

I. INTRODUCTION

Traffic in cities, traffic congestion and traffic jams are phenomena that has been extensively studied. One of the reasons in the interest for studying traffic is due to its importance in the wellbeing of people and its impact on the economy of societies because productivity and earnings are being affected by the conditions of traffic.

In [1] several sectors are identified that have interests in some aspect of traffic, for example, people living in the geographical zone under study, people living outside of the geographical zone under study, engineers working in traffic infrastructure, environmental, economic and political groups, to name a few. Also it is explained in [1] that the sectors mentioned in the previous sentence may not be interested in the same problems, but they may have conflicting goals.

There is a general consensus that population and economic growth, which are related to the number of vehicles in cities and the need to commute, is increasing so fast that cities's infrastructure cannot cope with the growth. Therefore, big cities are prone to traffic congestion problems.

Given the importance of traffic in the daily life of people, there is a considerable number of works in the literature that study the causes of traffic congestion and the different types of collapses; see [1], [2]. From these works, three possible causes of traffic congestion can be identified: (i) the inadequacy of streets to handle traffic flow, (ii) the inclusion of traffic elements that obstruct the flow, and (iii) deficient regulations. Therefore, geographical regions having a constant increment of traffic demand, and with no short-term plan for a significant improvement in infrastructure, need a rigorous study to identify what is required to improve traffic flow.

In order to optimize traffic flow, it is usually common to assume some parameters like number of sources for cars in a map and their corresponding appearance rate, number and position of traffic lights, etc. Given a set of such parameters, two important metrics that measure traffic flow are (1) time loss (TL) and (2) waiting time (WT). Theses two times are measured by taking their average over all cars during a time window. This way, a traffic flow through a map can be

This work is funded by CONACYT (National Council of Science and Technology of Paraguay) research project PINV15-873 "Mobile System for Counting and Characterization of Vehicles through Image Processing."

considered good when TL and WT are low. However, given a map and a traffic flow, how is it possible to calculate or estimate TL and WT? One way is to setup cameras in the geographical zone of interest, and another way is to run a traffic simulator. Setting up cameras to measure traffic flow can be costly in terms of money and logistics, and running a simulator can be computationally demanding taking too much computation time.

This work presents a way to simulate traffic using deep neural networks (DNNs). The goal is to obtain, with the highest possible precision, a good estimation of TL and WT for a given geographical map. Several architectures of DNNs were tested, and the best architectures for TL and WT exhibited a formidable improvement in computation time when compared against a freely available traffic simulator, with a decrease in precision of less that 10%. Then, the best architectures were also used in an application to find an optimal traffic light configuration using a genetic algorithm. This latter application was possible because of the huge improvements in computation time of the DNNs.

The rest of this work is organized as follow. Section II presents the current state-of-art in traffic simulation. Section III presents the main contributions of this work related to the simulation of traffic using DNNs. Section IV presents the genetic algorithm that uses the DNN found in this work to optimize traffic lights configurations. Finally, Section V presents the conclusions of this work with potential avenues for future research.

II. RELATED WORK

In [3] a survey on works related to car traffic prediction that are published in the main journals between the years 2004 and 2013 is presented. The authors of [3] concluded that, given the non-linear nature of traffic, one of the most utilized methods are based on neural networks. In particular, the use of deep learning is only reported in 1.8% of the papers in the survey. This number, however, can be due to the period of study of the survey as deep learning has been solidified as an area of research in recent years. Hence, deep learning, even though it is a more recent technique, it was already being applied to complex and relevant problems like car traffic with promising results.

In [4] a *deep belief network* is used to predict traffic speed in a street of Beijing. The authors showed that a deep belief network can have better precision than other more commonly used methods like *back-propagation neural networks* (BPNN) and *autoregressive integrated moving average* (ARIMA). A similar result is presented in [5] where different architectures of stacked autoencoders where studied and concluded that deep learning presents better precision than BPNNs or *support vector machines* (VPN). These works also mention that deep learning models are good at discovering the intricate patterns in the nature of traffic, however, there is no clear improvement in precision when the number of hidden layers or hidden nodes is increased.

In [6] a neural network is used to approximate the result of traffic simulations. In particular, the authors of [6] approximated the total waiting time of more than 40,000 cars in an area having 15 traffic lights. Furthermore, different sizes of neural networks were tested, like number of hidden layers and number of hidden nodes, thus concluding that size does not translate to an improvement in efficiency. This conclusion was reached by analyzing the mean squared error for each neural network. Nevertheless, these machine learning techniques are used not only for traffic prediction but to change the state of traffic with the goal of maximizing its flow. For example, in [6] a genetic algorithm is used to find an optimal configuration of traffic lights. Also in [7] a technique that changes traffic in order to maximize the flow is presented. This work presented an approach based on deep reinforcement learning for flow control in a relatively simple crossing (no turns) with one single traffic light. The authors of [7] compared their network against conventional reinforcement learning and showed that the deep reinforcement learning approach has better performance. In particular, the main conclusion of this latter work was that deep reinforcement learning produces an expected waiting time that is significantly less that the one obtained with other conventional reinforcement learning techniques.

III. TRAFFIC SIMULATION WITH DEEP NEURAL NETWORKS

A. Dataset Creation

In order to train the neural networks in this work, a dataset was created using SUMO [8]. The SUMO software have the following features:

- it allows the insertion of vehicles by type, time, and route;
- it is open source, portable and free;
- it can simulate any geographical zone as long as the map appears in open street maps[1];
- it allows the insertion of vehicles by type, time, and route;
- traffic lights cycles can be configured as needed.

SUMO has everything that is required to construct realistic simulation scenarios. SUMO, however, cannot execute simulations by batch, which is necessary to create the required amount of data for traffic simulation with neural networks; for example, in [6] 80,000 simulations were used. Moreover, the low-level control of SUMO requires many parameters to be set by hand, like the exact route for each vehicle in the simulator which needs to be specified explicitly for each vehicle. In order to automated this entire process, a *Training Data Generation Software* (TDGS) was created. The high-level workings of the TDGS is shown in Figure 1.

Each stage in the TDGS is explained as follows.

- *Amount of vehicles.* This stage reads an input file describing a list of cases to simulate, where each case describes the traffic demand and traffic lights configuration.
- *Vehicle demand file update.* This stage creates a file that defines the vehicle demand.

[1]https://www.openstreetmap.org/

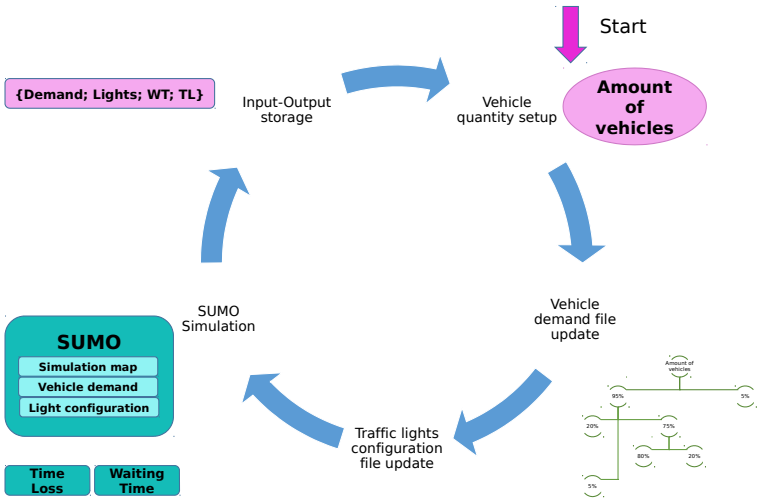

Figure 1. High-level workings of the Training Data Generation Software or TDGS.

- *Traffic lights configuration file update.* This stage creates a file that defines the traffic lights configuration.
- *SUMO simulation.* This stage uses the files in the previous two stages to run a SUMO simulation.
- *Input-Output storage.* This stage saves a file with the simulation results produced by SUMO together with the input data of the simulation. This file contains input-output pairs of SUMO saved as a CSV file. Aditionally, it saves the simulation time for each input-output pair.

For the purposes of training a neural network, each entry in the CSV file generated by the TDGS of Figure 1 is composed of 14 numbers, where the first 12 numbers is an input containing a traffic demand and traffic lights configuration and the other 2 remaining numbers are the waiting time and time loss of all simulated vehicles. These times are defined in [8] as follows.

- *Waiting Time (WT).* Average time spent standing still.
- *Time Loss (TL).* Average time lost due to driving slower than desired.

One important thing to note here is that WT is always less than or equal to TL.

From the 12 numbers, the first 7 numbers represent the traffic demand on the simulated geographical region, and each of these 7 numbers the represent the amount of vehicles inserted in 7 different points on the map for 25 minutes. See Figure 2 to see the particular map that was studied in this work.

The remaining 5 numbers in the array represent the traffic lights configuration in 5 different points on the map; see Figure 3.

This way, a simulation can be understood as a mapping from a set of arrays of 12 numbers to a set of pairs denoting TL and WT.

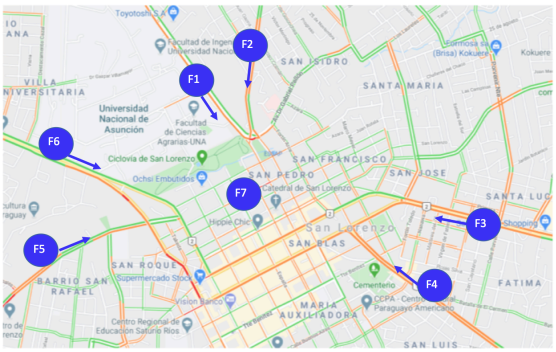

Figure 2. Map for the geographical zone of interest. Each point F1 to F7 are the insertion points of all cars that must be specified in SUMO.

When the TDGS ends its execution, a set of simulation results is obtained that is usable by data manipulation libraries like Pandas. At the same time, Pandas eases the integration with a training software written with TensorFlow.

B. Design and Training of a Network

The idea in this work is to replace SUMO with a DNN. Since a simulation by SUMO can be seen as a function mapping 12 numbers to a waiting time and time loss, it is expected that a DNN can "learn" this function.

Considering that the training of a neural network takes time and deciding for a network architecture is by trial-and-error, a software was developed that automates this entire process. From a finite pool of possible architectures, the software selects the neural network with the least validation error.

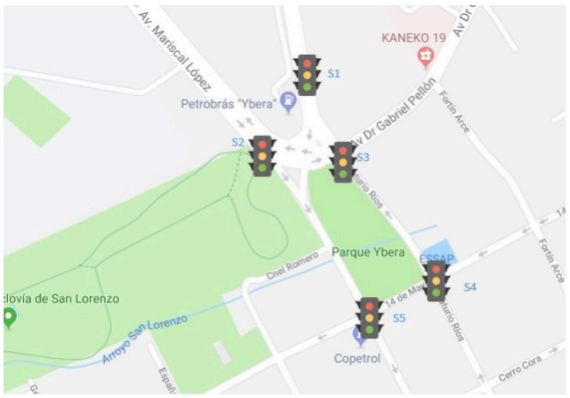

Figure 3. Map for the geographical zone of interest with the position of traffic lights.

Figure 4 shows a high-level description of the algorithm that trains neural networks by batch.

The entire process of Figure 4 can be explained as follows. The algorithm starts by receiving data obtained from the TDGS, which was explained in the previous subsection. The output data of the TDGS is divided in two groups.

- *Training data.* This set of data is used for the training of neural networks. Usually, an 80% of the available data is used for training according to the literature, which is the same criterion used in this work.
- *Validation data.* This set of data are not used during training, but it is used to assess the performance of the trained neural network.

The next step in the algorithm of Figure 4 "DNN Architectures" is to construct a list of neural network architectures to be evaluated. The parameters that were taken into account are:

- *Hidden units.* This is a list of numbers where each number indicates then amount of neurons for each hidden layer.
- *Optimizer.* This is the optimizer that will be used. In this work the *Adagrad* [9] optimizer was selected. The advantage of Adagrad is that variations in the learning rate can be introduced in order to adjust it to the application the user has in mind.
- *Dropout.* This is the chance of discarding a neuron during training. For this work, values 0 and 0.01 were tested.

After choosing a neural network architecture, the selected network is trained with the chosen parameters. Then, the results are saved in a file registering the performance of the network. This way, it is easy to find any overfitting in the model. After training all networks from the list of candidate architectures, the network with the best performance is selected.

In order to compare each neural network against each other and decide if a network is precise enough to replace SUMO, this work employed a set of widely use performance metrics that are common in the literature [10] which are (i) Mean Absolute Error (MAE), (ii) Root-Mean-Square Error (RMSE), and (iii) precision. These metrics show how good each network is for the given set of data.

One last thing to note is that in the algorithm of Figure 4 two different lists of neural network architectures were tested, one list for the time loss and another list for the waiting time. This way, each tested network is only trained with respect to one type of time. Details of the experiments realized on different DNN architectures are given in the appendix.

C. Results for the Neural Networks

After training and validating all networks using the algorithm of Figure 4, the best networks for TL and WT were selected. Table I present the networks with the best performance for each time feature.

For the networks shown in Table I, the results presented in Table II were obtained and compared against SUMO.

The results presented in tables I and II can be considered as promising, since a relatively small loss in precision translate to a large gain in execution time.

The importance of these results rely on the fact that many applications require a fast execution of traffic simulation. Hence, having a reliable and efficient simulator of traffic opens up many new avenues of research for optimizing traffic flow. In the next section, this work presents an application for optimizing the configuration of traffic lights using a genetic algorithm.

IV. OPTIMIZATION OF TRAFFIC LIGHTS

In this section, an application of a DNN for traffic prediction is presented. The goal is to find an optimal traffic light configuration that minimizes the time loss.

A. Design of a Genetic Algorithm

Recall that a genetic algorithm is an optimization algorithm that for a finite number of iterations applies so-called "genetic operations" to a finite set of feasible solutions. The set of feasible solutions is called a *population* and each feasible solution is called an *individual*. The set of genetic operations are selection, crossover, and mutation. The *fitness* of an individual is how desirable the individual is during selection.

In this work, the goal is to find a configuration of traffic lights in the geographical region of study that minimizes the time loss. Thus, the objective function (and hence the fitness of an individual) to be minimized is time loss, and an individual is any arbitrary configuration of traffic lights. The DEAP library for Python[2] was used.

The size of a population is 100 and the stopping criterion of the genetic algorithm is 20 iterations. The number 20 was selected because during the experiments of this work, it was observed that there is no considerable improvement of the population after 20 iterations. The initial population is created using randomly selected configurations of traffic lights. The

[2]https://github.com/deap

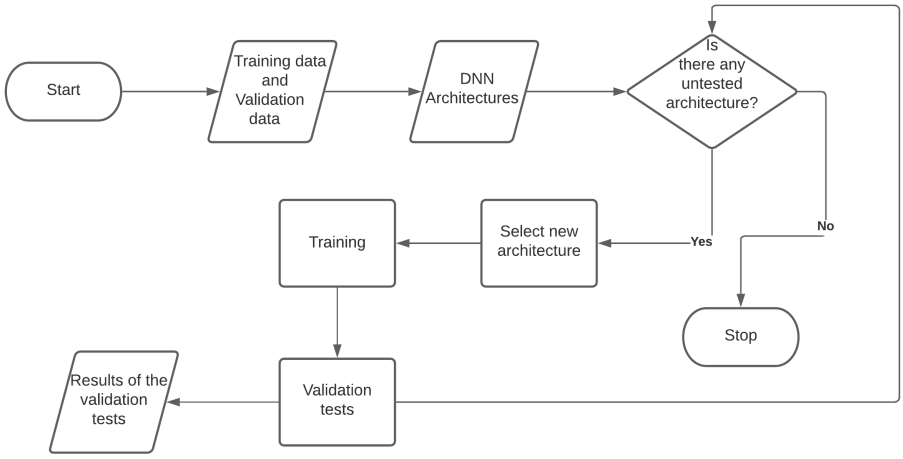

Figure 4. Algorithm for testing different DNN architectures.

Table I
DNN ARCHITECTURES WITH THE BEST PERFORMANCE.

	Network Architecture		Training	
Feature	**Hidden layers**	**Neurons**	**Learning rate**	**Dropout**
WT	12	100	0.001	0.01
TL	4	500	0.01	0.01

Table II
BEST DNN ARCHITECTURES AGAINST SUMO.

Metric	SUMO	WT DNN	TL DNN
Precision	100%	92.61%	95.44%
Execution time	39.68s	0.0177s	0.0177s

details of the genetic operations that are used in this study are given below.

- *Selection.* Individuals are selected using a tournament approach, where all individuals have the same probability of being selected. In each round of a tournament, for 100 rounds, two individuals are randomly selected with repetition and the one with the best fitness is promoted to next stage.
- *Crossover.* Two individuals are chosen for crossover following the order produced by the selection stage. Here, the *simulated binary crossover* was used invoking the function `cxSimulatedBinaryBounded(ind1,ind2,eta, low,up)` of DEAP, where `ind1` and `ind2` are the two individuals to be crossed, `eta=15`, `low=[10, 10, 10, 0, 0]` and `up= [600, 600, 600, 10, 10]`.
- *Mutation.* Here *polynomial mutation* was used which is implemented via the DEAP function `mutPolynomialBounded(individual, eta, low, up, indpb)` where `individual` is the individual to be mutated, `eta=15`, `low=[10, 10, 10, 0, 0]` and `up= [600, 600, 600, 10, 10]`, and `indpb=0.1` is the probability that an attribute of the individual is mutated.

The fitness of each individual is computed using the DNN selected for the time loss as reported in Table I. Using SUMO, instead of a DNN, would result in unreasonable computation times for the genetic algorithm because a SUMO simulation must be executed each time the fitness of an individual is computed. As mentioned before, the huge savings in computation time is compensated with a small loss in precision.

B. Results for the Genetic Algorithm

In order to test the genetic algorithm, several scenarios of traffic demand were studied. The traffic demand is specified by assigning a number to each source point F1 to F7 in Figure 2. The values of traffic demand used in this work are considered as *moderate traffic*.

Table III shows the different scenarios of traffic demand that were tested and the improvement in time loss using the genetic algorithm. The column `Gen0` of Table III shows for each scenario the average time loss for the initial population. Similarly, the column `Gen20` shows for each scenario the average time loss for the last population of the algorithm. The last column on the right is improvement in solution quality of the last population with respect to the initial population.

Table III
RESULTS OF THE GENETIC ALGORITHM FOR DIFFERENT SCENARIOS.

Traffic demand scenarios							Time Loss		Improvement
F01	F02	F03	F04	F05	F06	F07	Gen0	Gen20	
400	400	400	400	400	400	400	392	304	22.45%
300	100	100	100	100	100	100	253	221	12.65%
100	300	100	100	100	100	100	285	226	20.70%
100	100	300	100	100	100	100	246	225	8.54%
400	100	300	100	100	100	300	324	230	29.01%
400	200	300	200	200	200	300	350	250	28.57%
600	200	300	200	200	200	600	356	272	23.60%
100	300	300	200	600	200	400	352	285	19.03%
400	100	300	200	600	200	400	356	294	17.42%
150	150	150	150	150	150	150	300	228	24.00%

On average, there is an improvement in quality solution of approximately 20.6%. This result is promising since it shows that the proposed approach of using DNNs with genetic algorithms, or any other more sophisticated optimization algorithm, can be used to search for strategies to tackle traffic congestion.

V. CONCLUSIONS AND FUTURE WORKS

This work presented a method of simulating traffic using deep neural networks. Software tools like SUMO can be used also to simulate traffic, but simulation can take a very long time. Using neural networks this work showed that at the expense of some loss in precision, there is a incredible gain in computation speed. This fact allowed to use the neural network simulator inside a genetic algorithm that optimizes the average time loss of cars on a given geographical map. If a simulator like SUMO was used instead, the genetic algorithm approach would be very difficult to use due to the high computation times of the simulation.

The results presented in this work are in the initial stages, and further studies can help in a deep understanding of the methods presented here. A short list of interesting research paths is given below.

1) *Precision of DNNs for different geometries of maps.* Different map geometries carry different difficulties. Even though the results presented in this work are promising, a characterization of map geometries and the impact on the precision of DNNs will reveal for which type of maps the DNN approach works or not.
2) *Impact of traffic parameters on the architecture of DNNs.* In Section III, a study of different DNNs was presented in order to find the best network for the given traffic demand for the given map. The network architecture, however, can be affected by the map geometry, the traffic demand parameters, and the configuration of traffic lights. A study of how the number of hidden layers and number of neurons depend on the traffic could discover the power of DNNs as traffic simulators.
3) *Test different evolutionary algorithms.* This work presented a very simple evolutionary algorithm for the optimization of traffic lights. Since traffic is non-linear, any evolutionary algorithm could also work. A study comparing different metaheuristics like ant colony optimization, particle swarm optimization, etc, could give clues at the best optimization algorithm for traffic lights configuration.
4) *Multiobjective optimization of traffic lights.* To optimize traffic lights, the time loss was considered in this work, and the waiting time was discarded since it is always less than or equal to the time loss. However, during the experiments of this work, the simulation of vehicles showed that waiting time is independent of time loss, even though time loss is an upper bound on waiting time. It is an interesting problem to optimize simultaneously both times and find solutions using powerful metaheuristics.

REFERENCES

[1] J. C. Falcocchio and H. S. Levinson, *Road Rraffic Congestion: A Concise Guide*, vol. 7. Springer, 2015.

[2] A. Downs, *Stuck in Traffic: Coping with Peak-hour Traffic Congestion*. Brookings Institution Press, 2000.

[3] S. Suhas, V. V. Kalyan, M. Katti, B. A. Prakash, and C. Naveena, "A comprehensive review on traffic prediction for intelligent transport system," in *2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT)*, pp. 138–143, IEEE, 2017.

[4] Y. Jia, J. Wu, and Y. Du, "Traffic speed prediction using deep learning method," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1217–1222, IEEE, 2016.

[5] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE T. Intell. Transp.*, vol. 16, no. 2, pp. 865–873, 2014.

[6] P. Gora and M. Bardoński, "Training neural networks to approximate traffic simulation outcomes," in *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pp. 889–894, IEEE, 2017.

[7] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 247–254, 2016.

[8] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. WieBner, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2575–2582, IEEE, 2018.

[9] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, no. 61, pp. 2121–2159, 2011.

[10] J. Twomey and A. Smith, "Performance measures, consistency, and power for artificial neural network models," *Math. Comput. Model.*, vol. 21, no. 1-2, pp. 243–258, 1995.