IBB Traffic Graph Data: Benchmarking and Road Traffic Prediction Model

Eren Olug[1,3,4], Kiymet Kaya[2,3,4], Resul Tugay[5], and Sule Gunduz Oguducu[1,3]

[1] Istanbul Technical University, Department of Artificial Intelligence and Data Engineering, Istanbul, Turkey
[2] Istanbul Technical University, Department of Computer Engineering, Istanbul, Turkey
[3] ITU AI Research and Application Center, Istanbul, Turkey
[4] BTS Group, Istanbul, Turkey
[5] Gazi University, Department of Computer Engineering, Ankara, Turkey
Email: olug20@itu.edu.tr, kayak16@itu.edu.tr, resultugay@gazi.edu.tr, sgunduz@itu.edu.tr

Abstract—Road traffic congestion prediction is a crucial component of intelligent transportation systems, since it enables proactive traffic management, enhances suburban experience, reduces environmental impact, and improves overall safety and efficiency. Although there are several public datasets, especially for metropolitan areas, these datasets may not be applicable to practical scenarios due to insufficiency in the scale of data (i.e. number of sensors and road links) and several external factors like different characteristics of the target area such as urban, highways and the data collection location. To address this, this paper introduces a novel IBB Traffic graph dataset as an alternative benchmark dataset to mitigate these limitations and enrich the literature with new geographical characteristics. IBB Traffic graph dataset covers the sensor data collected at 2451 distinct locations. Moreover, we propose a novel Road Traffic Prediction Model that strengthens temporal links through feature engineering, node embedding with GLEE to represent inter-related relationships within the traffic network, and traffic prediction with ExtraTrees. The results indicate that the proposed model consistently outperforms the baseline models, demonstrating an average accuracy improvement of 4%.

Index Terms—road traffic prediction, graph representation learning, node embeddings

I. INTRODUCTION

The importance of intelligent transportation systems (ITSs) has rapidly increased as they enhance travel efficiency, reduce traffic incidents, and improve safety. Traffic prediction involves predicting future traffic behavior in a given area according to current and past traffic conditions. This information is crucial for multiple purposes, such as reducing congestion, optimizing transportation systems, and enhancing road safety.

With the growing capabilities of road sensor devices, various information about traffic networks, such as flow and speed, can be collected and prepared for several downstream tasks not only for traffic predicting, including forecasting of traffic flow, travel time, and traffic density, but also other real-world applications such as travel demand prediction for ride-hailing services, autonomous vehicles, intersection management, parking management, urban planning, and transportation safety [1]. Traditionally, traffic prediction has utilized methods like rule-based models and time-series analysis. However, these approaches often fall short of capturing the full complexity, variability of traffic patterns in graph data.

Traffic flow prediction for a specific road is a difficult problem because it is influenced by many factors, including the day, time of day, weather conditions, and the traffic flow of neighboring roads. Thus, traffic prediction requires generating data from the non-Euclidean domain, necessitating the use of graph data structures to represent complex spatial relationships and interdependencies between objects [2]. This task is particularly challenging due to the spatiotemporal nature of traffic network data, which encapsulates both intricate spatial relationships and dynamic temporal dependencies [3], [4].

Graph Neural Networks (GNNs) can directly process the graph structure (i.e., adjacency information) along with node (sensors in roads) features. Numerous techniques for extracting features from graphs [5]–[8] have been shown to be effective in handling graph structures, including their adjacency information. These methods enable the extraction of node embeddings of the graph in Euclidean space.

Publicly available road traffic prediction datasets differ in several ways (1) size: the number of sensors or data points, (2) temporal coverage: how long period the data covers, (3) sensor metadata: information about the sensors (e.g. location, type), and (4) road type: whether the data is collected on highways or urban streets. One major limitation in many popular datasets, like METR-LA [9], is their focus on only highways. This focus results in the lack of ability to recognize urban traffic networks successfully. Another limitation is that these datasets are often scale-limited compared to real-world road networks, making it harder to train complex forecasting models. For instance, METR-LA and PeMS-BAY [10], two of the most widely used datasets, contain only 207 and 325 sensors, respectively. Additionally, only a few of the datasets [10] cover a long enough temporal period, which is crucial in capturing long-term traffic patterns. Considering the impact of data quality on model performance, these limitations reduce the effectiveness of traffic prediction models.

In this study, we aim to enrich the literature by introducing the IBB Traffic graph dataset, which covers 2,451 sensors located in Istanbul, Turkey, including both urban and highway roads. Given the features collected by sensors, this dataset can be utilized not only for predicting road traffic anomalies but also for forecasting speed, flow, and travel time. Moreover, we propose a novel Road Traffic Prediction Model that strengthens temporal links through feature engineering, node embedding with GLEE to represent inter-related relationships within the traffic network, and traffic prediction with ExtraTrees.

The main contributions of our study can be summarized as follows:

- **A novel IBB traffic graph dataset** comprises sensor

traffic data collected from 2,451 distinct locations over a four-year period with measurements taken at one-hour intervals.

- **A novel Road Traffic Prediction Model** that strengthens temporal links through feature engineering, node embedding with GLEE to represent inter-related relationships within the traffic network, and traffic prediction with ExtraTrees.

The rest of the paper is organized as follows. Section II presents related works on traffic congestion prediction problem. Section III gives details of the IBB traffic graph data generation process. Section IV gives details of the proposed methodology. Section V presents the experimental results of the study. Lastly, Section VI concludes the paper.

II. LITERATURE REVIEW

Extreme Gradient Boosting (XGBoost) is proposed for higher resolution traffic state estimation using the origin-destination relationship of segment flow data between upstream and downstream on the highway [11]. The performance of machine learning methods such as linear regression, Random Forest (RF), and ARIMA is evaluated for traffic congestion prediction, which is one of the vital components of Intelligent Transportation Systems in smart cities [12]. The growth of big data makes it possible to study traffic congestion issues through data from sensors in various locations. Considering rainfall data as a traffic congestion indicator, a stacked generalization model suitable for big and complex data using RF, CatBoost, and XGBoost algorithms is proposed [13]. LightGBM model proposed for Freeway Short-Term Travel Time Prediction is proved to be more accurate and has a shorter model training time than KNN and Gradient Boosting Machine (GBM) [14].

Models trained and evaluated on widely used traffic datasets may not be applicable to practical scenarios due to the limited scale of these datasets compared to real-world traffic networks [10]. In other words, widely used datasets such as PeMS03, PeMS04, PeMS07, PeMS08, METR-LA, and PeMS-BAY [9], [15] contain a low number of sensors, resulting in a small road network with an insufficient number of nodes. In this study, we enrich the literature by introducing the IBB Traffic graph dataset, which covers 2451 sensors located in Istanbul, including both urban and highway roads.

III. IBB GRAPH-BASED ROAD NETWORK DATASET

Traffic in Istanbul is notable due to the city's unique characteristics, including its distinctive geographical features and its status as a metropolitan area with over five million vehicles [16]. The city spans two large landmasses, each located on a different continent (Asia and Europe), separated by the Bosporus Strait. In addition to intra-continental traffic flow, millions of people travel between continents daily using three different bridges connecting these continents.

The proposed IBB Traffic graph dataset encompasses four years of traffic data, including features such as traffic state (e.g., flow and speed) and sensor information. The traffic data-collecting sensors are strategically positioned at approximately equal intervals throughout the city, covering both highways and urban areas. In the IBB Traffic graph dataset, sensor measurement frequency is one hour, yielding 24 measurements

Fig. 1. Istanbul Road Traffic Network

per day for each of the 2451 sensors. Since edge connections are not explicitly defined, we connected two nodes with an unweighted edge if the corresponding sensors were adjacent to each other. This approach results in a total of 6667 edges and an average degree of approximately 2.7 in the road network. The topology of the resulting data is visualized in geolayout representation in Figure 1 to provide an insight into the IBB Traffic network.

In the graph representation of the IBB Traffic dataset, all features are encoded as node features meaning the edges are non-attributed. The node features consist of two categories: traffic flow features and sensor features. As the name implies, traffic features are collected from traffic flow measured by sensors in that time range such as *minimum speed*, *maximum speed*, *average speed*, and *number of passing vehicles*. On the other hand, sensor features include spatial information about sensors such as *continent*, *district*, *whether the sensor is on a highway or not*, and *the population density* around the sensor.

IV. METHODOLOGY

The proposed Road Traffic Prediction Model is illustrated in Figure 2. The model benefits of traffic flow features: average speed, vehicle count, minimum speed, and maximum speed, and sensor features encompass geographical categorizations like continents, districts, highways/urban areas, and population density, which are shown in maps. With feature engineering study, previous week and hour traffic densities are gathered. Moreover, node embeddings are obtained with the GLEE method. All of the features are then processed and combined by the ExtraTrees classifier to distinguish between normal and congestion traffic. Next, we give details of the node embedding method GLEE and traffic congestion classification method ExtraTrees. The implementation details and the code repository of the proposed model are available here[1].

A. Node Embedding: GLEE

GLEE [5] is closely related to the Laplacian Eigenmaps (LE) method. LE constructs node embeddings based on the spectral properties of the Laplacian matrix of the graph. However, while LE aims to minimize the distance between similar nodes based on the spectral properties of the Laplacian matrix, GLEE disposes of the distance-minimization assumption and uses the Laplacian matrix to find embeddings with geometric properties, by leveraging the simplex geometry of the graph.

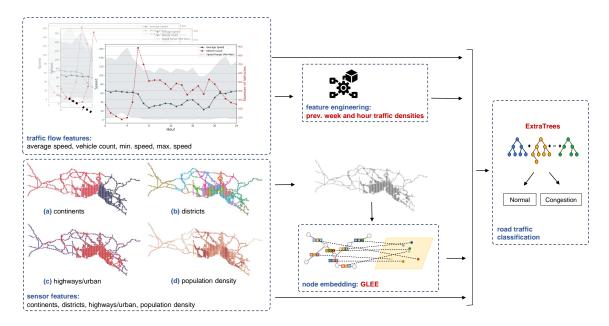[1]https://github.com/erenolg/IBBTraffic

Fig. 2. Road Traffic Prediction Model

Let Λ be the diagonal matrix having eigenvalues of $\mathbf{L}$ as entries, GLEE solves a distance maximization problem given in Equation 1, where the objective function is $\mathrm{tr}(\mathbf{Y}^T\mathbf{L}\mathbf{Y})$ under the constraint $\mathbf{Y}^T\mathbf{Y} = \Lambda$.

$$\arg\max_{\mathbf{Y}\in\mathbb{R}^{n\times d}} \mathrm{tr}(\mathbf{Y}^T\mathbf{L}\mathbf{Y}) \quad \text{s.t.} \quad \mathbf{Y}^T\mathbf{Y} = \Lambda \quad (1)$$

More concretely, given a set of data points $\{x_1, x_2, \ldots, x_n\}$, the first step in Geometric Laplacian Eigenmap Embedding (GLEE) is to construct a graph $G = (V, E)$ where each node represents a data point. The edge weight matrix $\mathbf{A}$, is typically computed using a Gaussian kernel as in Equation 2. Next, the degree matrix $\mathbf{D}$ is computed, which is a diagonal matrix represents the sum of the weights of the edges connected to node i.

$$A_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right). \quad (2)$$

Given the weight matrix $\mathbf{A}$ and the degree matrix $\mathbf{D}$, we then compute the $\mathbf{L}$, defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. The core of GLEE involves solving the generalized eigenvalue problem $\mathbf{L}\mathbf{y} = \lambda\mathbf{D}\mathbf{y}$, where $\mathbf{y}$ are the eigenvectors and λ are the eigenvalues. The solution to this eigenvalue problem provides a set of eigenvectors $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m$ corresponding to the smallest eigenvalues. Finally, the data points x_i are embedded into a lower-dimensional space by using the eigenvectors as coordinates $y_i = (y_i^{(1)}, y_i^{(2)}, \ldots, y_i^{(m)})$, where $y_i^{(j)}$ is the i-th component of the j-th eigenvector. This process results in a low-dimensional representation of the original high-dimensional data, preserving the geometric structure of the data as captured by the graph.

B. Road Traffic Prediction: ExtraTrees

ExtraTrees is an ensemble model based on decision trees. ExtraTrees differ from standard bagging models in terms of tunning of *best splitting* hyperparameter. ExtraTrees chooses the optimum split randomly which makes it much faster.

V. EXPERIMENTAL RESULTS

We conducted our experiments using four node embedding methods, three proximity-preserving (Node2Vec, GLEE, NetMF), one structural node-level embedding (GraphWave), and seven classifiers. Next, we give details of these models used in the experiments and road traffic prediction results obtained.

Node2Vec [17] algorithm learns representations of nodes (i.e. sensors) via random walk procedure. The algorithm creates node embeddings of the graph following a flexible neighborhood sampling strategy that trades off between Breadth-First Search (BFS) and Depth-First Search (DFS). To illustrate, starting from a source node u, Node2Vec simulates a random walk of length l. Initially, $c_0 = u$ represents the starting node, and c_i denotes the i th node in the walk. The nodes c_i are generated based on the probability distribution in Equation 3. In this equation, π_{xv} is the unnormalized transition probability between nodes v and x, and Z is the normalizing constant.

$$P\left(c_i = x \mid c_{i-1} = v\right) = \left\{ \begin{array}{cc} \frac{\pi_{vx}}{Z} & \text{if } (v,x) \in E \\ 0 & \text{otherwise} \end{array} \right\} \quad (3)$$

A 2nd order random walk on a graph is defined by two parameters p (return parameter) and q (in-out parameter) which guide the walker in deciding the next step. Consider a scenario in which the walker has just taken a step from node t to node v. At this point, the walker evaluates the transition probabilities π_{vx} on edges (v, x) leading from v. The transition probability here is unnormalized and can be represented as $\pi_{vx} = \alpha_{pq} \cdot w_{vx}$ where the α_{pq}, which defined in Equation 4, is the length of the shortest path between nodes t and x.

$$\alpha_{pq}(t,x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases} \quad (4)$$

TABLE I
TRAFFIC CONGESTION CLASSIFICATION RESULTS

Classification Model	wo/ feature enginnering					w/ feature enginnering				
	Acc.	Precision	Recall	F1	AUROC	Acc	Precision	Recall	F1	AUROC
LR	0.7739	**0.6250**	0.2066	0.3105	0.8005	**0.9572**	0.9310	**0.8925**	**0.9113**	**0.9920**
KNN	0.7596	0.5180	**0.3553**	0.4215	0.7491	0.9490	0.9000	**0.8925**	0.8962	0.9625
RF	0.7820	0.5972	**0.3553**	**0.4456**	0.8396	0.9551	0.9230	**0.8925**	0.9075	0.9829
XGBoost	**0.7841**	0.6119	0.3384	0.4361	0.8410	0.9551	0.9304	0.8843	0.9067	0.9891
CatBoost	**0.7841**	0.6119	0.3384	0.4361	0.8418	0.9531	0.9224	0.8843	0.9029	0.9906
LightGBM	**0.7841**	0.6119	0.3384	0.4361	**0.8421**	0.9470	0.9203	0.8595	0.8888	0.9903
ExtraTree	**0.7841**	0.6119	0.3384	0.4361	0.8400	0.9551	**0.9380**	0.8760	0.9059	0.9868

TABLE II
EMBEDDING AIDDED TRAFFIC CONGESTION CLASSIFICATION RESULTS

Embedding Model	Classification Model	wo/ feature enginnering					w/ feature enginnering				
		Acc.	Precision	Recall	F1	AUROC	Acc.	Precision	Recall	F1	AUROC
Node2Vec	LR	0.7657	0.5517	0.2644	0.3575	0.8085	0.9633	0.9401	0.9090	0.9243	0.9912
	KNN	0.8329	0.6560	0.6776	0.6666	0.8694	0.8655	0.7165	0.7520	0.7338	0.9059
	RF	**0.8513**	**0.7222**	0.6446	**0.6812**	0.9145	0.9592	0.9469	0.8843	0.9145	0.9856
	XGBoost	0.8492	0.7117	0.6528	0.6810	0.9150	0.9613	0.9322	0.9090	0.9205	0.9901
	CatBoost	0.8492	0.7156	0.6446	0.6782	**0.9183**	0.9613	0.9473	0.8925	0.9191	0.9904
	LightGBM	0.8391	0.6981	0.6115	0.6519	0.9081	0.9613	0.9250	**0.9173**	0.9211	0.9913
	ExtraTree	0.8411	0.6869	0.6528	0.6694	0.9056	0.9613	0.9473	0.8925	0.9191	0.9897
GLEE	LR	0.7739	0.6250	0.2066	0.3105	0.8016	0.9572	0.9310	0.8925	0.9113	**0.9922**
	KNN	0.8309	0.6727	0.6115	0.6406	0.8321	0.9490	0.9000	0.8925	0.8962	0.9650
	RF	0.8268	0.6525	0.6363	0.6443	0.8798	0.9490	0.9137	0.8760	0.8945	0.9868
	XGBoost	0.8146	0.6293	0.6033	0.6160	0.8798	0.9429	0.9043	0.8595	0.8813	0.9870
	CatBoost	0.8126	0.6435	0.5371	0.5855	0.8931	0.9531	0.9224	0.8843	0.9029	0.9901
	LightGBM	0.8289	0.6554	0.6446	0.6500	0.8928	0.9511	0.9145	0.8843	0.8991	0.9885
	ExtraTree	0.8370	0.6614	**0.6942**	0.6774	0.8603	**0.9653**	**0.9482**	0.9090	**0.9282**	0.9877
NETMF	LR	0.8228	0.7073	0.4793	0.5714	0.8405	0.9572	0.9464	0.8760	0.9098	0.9910
	KNN	0.8350	0.6666	0.6611	0.6639	0.8670	0.9307	0.8320	0.9008	0.8650	0.9560
	RF	0.8411	0.7009	0.6198	0.6578	0.8833	0.9511	0.9145	0.8843	0.8991	0.9860
	XGBoost	0.8431	0.6964	0.6446	0.6695	0.8947	0.9490	0.9000	0.8925	0.8962	0.9875
	CatBoost	0.8370	0.7113	0.5702	0.6330	0.9075	0.9490	0.9137	0.8760	0.8945	0.9901
	LightGBM	0.8411	0.7128	0.5950	0.6486	0.9011	0.9551	0.9230	0.8925	0.9075	0.9895
	ExtraTree	0.8329	0.6695	0.6363	0.6525	0.8726	0.9592	0.9316	0.9008	0.9159	0.9889
GraphWave	LR	0.7739	0.6250	0.2066	0.3105	0.8038	0.9572	0.9310	0.8925	0.9113	0.9922
	KNN	0.7922	0.5887	0.5206	0.5526	0.8005	0.9490	0.9000	0.8925	0.8962	0.9625
	RF	0.7881	0.5794	0.5124	0.5438	0.8563	0.9592	0.9391	0.8925	0.9152	0.9863
	XGBoost	0.8044	0.6213	0.5289	0.5714	0.8557	0.9531	0.9224	0.8843	0.9029	0.9885
	CatBoost	0.7983	0.6100	0.5041	0.5520	0.8707	0.9613	0.9473	0.8925	0.9191	0.9900
	LightGBM	0.8309	0.6862	0.5785	0.6278	0.8836	0.9531	0.9298	0.8760	0.9021	0.9898
	ExtraTree	0.7881	0.5794	0.5124	0.5438	0.8530	0.9633	0.9478	0.9008	0.9237	0.9834

The parameters p and q determine how the random walk behaves, like either BFS or DFS. Parameter p influences the likelihood of revisiting a node immediately after leaving it. Higher values of p reduce this likelihood, which means higher values for p encourage the walker to make more exploration. The other parameter q determines whether the walker searches for inward or outward nodes. The two parameters are intertwined, meaning the value of one does not alone tell about the behavior of the random walk.

NetMF [18] unifies various network embedding models, namely DeepWalk, LINE, PTE, and Node2Vec into an implicit matrix factorization framework, showing their theoretical connections to graph Laplacians. DeepWalk generates random walks on the network similar to how sentences are formed in a language. The NetMF aims to maximize the probability of observing these node sequences, which can be represented as matrix factorization of the transition probability matrix $\mathbf{M}$ in Equation 5, where U and V are the node embeddings. LINE (Large-scale Information Network Embedding) focuses on preserving both first-order and second-order proximities. First-order proximity ensures that neighbor nodes or nodes with strong connections are closer to each other in the embedding space, while second-order proximity can be interpreted as nodes sharing the same neighbors are more likely to be similar.

$$\min_{U,V} \|M - UV^\top\|^2, \tag{5}$$

$$\min_{U,V} \sum_t \sum_{(i,j)\in E_t} w_{ij}^t \|u_i^t - v_j^t\|^2, \tag{6}$$

$$\min_{U,V} \|M_{\text{node2vec}} - UV^\top\|^2. \tag{7}$$

$$\min_{U,V} \|M - UV^\top\|^2, \qquad (8)$$

PTE (Predictive Text Embedding) is an extension of LINE in heterogeneous networks, optimizing a similar objective for each type of edge and combining them as in Equation 6 where t indexes different types of edges. Node2vec extends DeepWalk by introducing a biased random walk to capture homophily and structural equivalence. All these methods can be unified under the general matrix factorization framework in Equation 8 where M represents different transition probability matrices or proximity matrices depending on the method used.

GraphWave [8] is used to learn the neighborhood of each node in a network using a low-dimensional embedding by leveraging heat wavelet diffusion patterns. Let $\mathbf{U}$ be the eigenvector matrix from the decomposition of the unnormalized graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^T$, Λ is a diagonal matrix of eigenvalues λ and $\lambda_1 < \lambda_2 \leq \ldots \leq \lambda_N$ are the eigenvalues of $\mathbf{L}$. Here, $\mathbf{D}$ is the diagonal matrix whose entries are the degrees of each node, and $\mathbf{A} = (a_{ij})$ is the edge weight matrix such that a_{ij} represents the weight of the edge between nodes i and j. Let g_s be a filter kernel with scaling parameter s. The spectral graph wavelet of g_s is the signal resulting from the modulation in the spectral domain of a Dirac signal centered around node a. The spectral graph wavelet Ψ_a is given as an N-dimensional vector as in Equation 9, where $\delta_a = \Vdash(a)$ is the one-hot vector of node a. Spectral graph wavelets are based on the relationship between the temporal frequencies of a signal and the Laplacian's eigenvalues.

$$\Psi_a = \mathbf{U}\,\mathrm{Diag}\left(g_s\left(\lambda_1\right), \ldots, g_s\left(\lambda_N\right)\right)\mathbf{U}^T\delta_a \qquad (9)$$

$$\phi_X(t) = \mathbb{E}\left[e^{itX}\right] \qquad (10)$$

$$\Psi_a(t) = \frac{1}{N}\sum_{m=1}^{N} e^{it\Psi_{ma}} \qquad (11)$$

$$\chi_a = \left[\mathrm{Re}\left(\phi_a\left(t_i\right)\right), \mathrm{Im}\left(\phi_a\left(t_i\right)\right)\right] \qquad (12)$$

For every node a, GraphWave computes a $2d$-dimensional vector χ_a. To obtain diffusion patterns of every node, spectral graph wavelets are used. Diffusion patterns are stored in a matrix Ψ. Ψ is a $N x N$ matrix, where the a-th column is the spectral graph wavelet for a heat kernel centered at node a. GraphWave embeds spectral graph wavelet coefficient distributions into $2d$ dimensional space by calculating the characteristic function for each node's coefficients Ψ_a and samples it at d evenly spaced points. The characteristic function of a probability distribution X is as in Equation 10. For a given node a and scale s, the empirical characteristic function of Ψ_a is defined as in Equation 11. Embedding χ_a of node a is obtained by sampling the parametric function in Equation 10 at d evenly spaced points $t_1 \ldots t_d$ and concatenating the values as in Equation 12.

Logistic Regression (LR) uses the sigmoid function $\sigma(x) = 1/(1 + e^{-x})$ to transform the weighted sum of input features into a probability value ranging between zero and one.

K-Nearest Neighbor (KNN) classifies data points based on their proximity to other data points.

Random Forest (RF) is a bootstrap ensemble model. It creates several decision trees on data samples and then selects the best solution using voting.

XGBoost is an implementation of gradient boosting. The key idea behind the XGBoost is the improvement of speed and performance using the reasons behind the good performance such as regularization and handling sparse data.

LightGBM is a histogram-based gradient boosting algorithm reducing the computational cost by converting continuous variables into discrete ones. Since the training time of the decision trees is directly proportional to the number of computations, hence the number of splits, LightGBM provides a shorter model training time and efficient resource utilization.

Catboost is a gradient-boosting ML algorithm known for its high predictive accuracy and speed. It employs techniques such as ordered boosting and oblivious trees to handle various data types effectively and mitigate overfitting.

A. Traffic Congestion Prediction Results

In this study, we approached the traffic congestion prediction problem as a binary classification task, where the output for each sensor node determines whether congestion may occur or not. A straightforward and commonly used method for labeling involves directly using average speed values, as shown in Equation 13. However, given the narrow streets in urban areas where the average speed is low but congestion is absent, we conducted our experiments using an alternative formulation. This formulation also takes into account the number of passing vehicles, as represented in Equation 14. The function C gives the proper label on congestion for each node in a timestamp where the s is the speed and v is the number of vehicles. Additionally, σ is the sigmoid function mapping the output between 0 and 1 and the τ is the threshold value ranging between 0 and 1.

$$C(s) = \begin{cases} 0, & \text{if } \frac{1}{e^s} < \tau \\ 1, & \text{otherwise} \end{cases} \qquad (13)$$

$$C(s, v) = \begin{cases} 0, & \text{if } \sigma\left(\frac{s}{v}\right) < \tau \\ 1, & \text{otherwise} \end{cases} \qquad (14)$$

The results obtained with various classifier models without adding node embedding vectors to the data are presented in Table I, while the results with node embedding are presented in Table II. In both Table I and Table II, the value of feature engineering studies was also queried. The values highlighted in Table I indicate the most accurate value for the relevant metric.

Comparing the 'without feature engineering' and 'with feature engineering' cases, it is seen that the traffic density one hour ago and the traffic density one week ago features obtained via feature engineering studies are quite effective. On the other hand, in Table II, the light green color indicates that node embeddings provide a performance improvement compared to the result in Table I for that metric, classifier and w/wo feature cases, while dark greens indicate the highest scores in that category. When the results in Table I and Table II are analyzed together, for the scenario 'without feature engineering', the approach of adding node embedding vectors to the data improves the prediction performance regardless of the classifier and node embedding methods. Considering all

the findings together, it is seen that the prominent approach (with Acc.= 0.9653, Precision= 0.9482, Recall= 0.9090, F1= 0.9282, AUROC= 0.9877) is the one that strengthens temporal links through feature engineering, node embedding with GLEE to represent inter-related relationships within the traffic network and road traffic congestion prediction with ExtraTrees.

VI. CONCLUSION AND FUTURE WORKS

In this study, we propose a novel Road Traffic Prediction Model that effectively leverages traffic flow features—such as average speed, vehicle count, min. speed, and max. speed—and sensor features like geographical categorizations (continents, districts, highways/urban areas, and population density), which are visualized through maps. The inclusion of temporal data w/feature engineering studies, such as traffic densities from the previous week and hour, further enhances prediction accuracy. Node embeddings obtained via the GLEE, combined with the ExtraTrees classifier, enable the distinction between normal and congested traffic more accurately. Analysis of results shows significant performance improvements when using feature engineering and node embeddings, highlighting the importance of these techniques in traffic prediction. For future work, we plan to work on traffic prediction for dynamic graphs.

REFERENCES

[1] S. Rahmani, A. Baghbani, N. Bouguila, and Z. Patterson, "Graph neural networks for intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, pp. 8846–8885, 2023.

[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.

[3] M. Jin, H. Y. Koh, Q. Wen, D. Zambon, C. Alippi, G. I. Webb, I. King, and S. Pan, "A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection," 2023.

[4] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, and B. Yin, "Deep learning on traffic prediction: Methods, analysis, and future directions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4927–4943, 2021.

[5] L. Torres, K. S. Chan, and T. Eliassi-Rad, "Glee: Geometric laplacian eigenmap embedding," *Journal of Complex Networks*, vol. 8, no. 2, Mar. 2020. [Online]. Available: http://dx.doi.org/10.1093/comnet/cnaa007

[6] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," 2017.

[7] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '14. ACM, Aug. 2014. [Online]. Available: http://dx.doi.org/10.1145/2623330.2623732

[8] C. Donnat, M. Zitnik, D. Hallac, and J. Leskovec, "Learning structural node embeddings via diffusion wavelets," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '18. ACM, Jul. 2018. [Online]. Available: http://dx.doi.org/10.1145/3219819.3220025

[9] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2018. [Online]. Available: https://arxiv.org/abs/1707.01926

[10] X. Liu, Y. Xia, Y. Liang, J. Hu, Y. Wang, L. Bai, C. Huang, Z. Liu, B. Hooi, and R. Zimmermann, "Largest: A benchmark dataset for large-scale traffic forecasting," 2023.

[11] B. Sun, T. Sun, and P. Jiao, "Spatio-temporal segmented traffic flow prediction with anprs data based on improved xgboost," *Journal of Advanced Transportation*, vol. 2021, no. 1, p. 5559562, 2021.

[12] M. Bai, Y. Lin, M. Ma, P. Wang, and L. Duan, "Prepct: Traffic congestion prediction in smart cities with relative position congestion tensor," *Neurocomputing*, vol. 444, pp. 147–157, 2021.

[13] M.-D. Tran and N. M. Saleem, "Predicting high-risk congestion areas during heavy rain using multi prediction model and maximum periodic frequent pattern algorithms," in *Proceedings of the 3rd ACM Workshop on Intelligent Cross-Data Analysis and Retrieval*, 2022, pp. 27–31.

[14] F. Wang, H. Cheng, H. Dai, and H. Han, "Freeway short-term travel time prediction based on lightgbm algorithm," in *IOP Conference Series: Earth and Environmental Science*, vol. 638, no. 1. IOP Publishing, 2021, p. 012029.

[15] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, pp. 914–921, Apr. 2020.

[16] "TUIK," *https://data.tuik.gov.tr/Bulten/Index?p=Motorlu-Kara-Tasitlari-Aralik-2023-49432*.

[17] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," 2016.

[18] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, ser. WSDM 2018. ACM, Feb. 2018. [Online]. Available: http://dx.doi.org/10.1145/3159652.3159706