




Article

# Optimized CapsNet for Traffic Jam Speed Prediction Using Mobile Sensor Data under Urban Swarming Transportation

Hendrik Tampubolon <sup>1</sup>, Chao-Lung Yang <sup>2,\*</sup> , Arnold Samuel Chan <sup>2</sup>, Hendri Sutrisno <sup>2</sup>   
and Kai-Lung Hua <sup>1</sup> 

<sup>1</sup> Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei 106, Taiwan; D10715806@mail.ntust.edu.tw (H.T.); hua@mail.ntust.edu.tw (K.-L.H.)

<sup>2</sup> Department of Industrial Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan; M10601803@mail.ntust.edu.tw (A.S.C.); D10701802@mail.ntust.edu.tw (H.S.)

\* Correspondence: clyang@mail.ntust.edu.tw; Tel.: +886-2-2730-3621

Received: 30 October 2019; Accepted: 27 November 2019; Published: 29 November 2019



**Abstract:** Urban swarming transportation (UST) is a type of road transportation where multiple types of vehicles such as cars, buses, trucks, motorcycles, and bicycles, as well as pedestrians are allowed and mixed together on the roads. Predicting the traffic jam speed under UST is very different and difficult from the single road network traffic prediction which has been commonly studied in the intelligent traffic system (ITS) research. In this research, the road network wide (RNW) traffic prediction which predicts traffic jam speeds of multiple roads at once by utilizing citizens' mobile GPS sensor records is proposed to better predict traffic jam under UST. In order to conduct the RNW traffic prediction, a specific data preprocessing is needed to convert traffic data into an image representing spatial-temporal relationships among RNW. In addition, a revised capsule network (CapsNet), named OCapsNet, which utilizes nonlinearity functions in the first two convolution layers and the modified dynamic routing to optimize the performance of CapsNet, is proposed. The experiments were conducted using real-world urban road traffic data of Jakarta to evaluate the performance. The results show that OCapsNet has better performance than Convolution Neural Network (CNN) and original CapsNet with better accuracy and precision.

**Keywords:** traffic jam prediction; urban swarming transportation; capsule network; convolution neural network; smart city

## 1. Introduction

The term “smart city” has been defined by IBM [1] to indicate a smart city that utilizes information and communication technology to analyze and integrate the data into core systems for running the city. The key enabler of the smart city depends on the connected devices and how the collected data, generated through the Internet of Things (IoT) sensors [2], is used. As the volume and variety of data offered by the IoT keeps increasing exponentially, how to utilize the data and transform it to knowledge for a smart city are crucial tasks for modern civilization.

A large amount of data collected from speed sensors or surveillance camera systems have been used to monitor traffic conditions on roads in an intelligence traffic system (ITS) domain. The most common detection technologies are loop detector, road-side cameras, and on-board equipment [3]. Lv, Yisheng, et al. utilized California's freeway traffic detector station data to predict traffic flow [4]. Zhao, Chen also utilized Beijing's ring road observation station, which is equipped with cameras, induction coils, and velocity radars to make short-term forecasts of traffic volumes [5]. Similarly,

closed circuit television camera (CCTV), laser detector, and loop detector data were utilized by Kim and Hong to predict the traffic flow in Suwon city intersection roads in South Korea [6]. All of the mentioned research works relied on data from fixed traffic detectors which are costly infrastructures of a transportation system. Therefore, how to obtain less costly and more accurate traffic flow information is an important issue in this research area.

Limited budgets for ITS implementation in developing countries where the transportation infrastructure is relatively straggly hinders the detection of traffic conditions. Fortunately, the telecommunication infrastructure in these developing counties comparatively prevails. The use of smartphones, in fact, provides a sufficient source of traffic data by tracing the global positioning system (GPS) information. For example, the government of Jakarta city, the capital city of Indonesia, has taken advantage of the data feed of citizen's smartphones to aggregate the data for traffic usage by collaborating with Waze® [7], which is a popular navigation software installed on citizens' smartphone. Instead of having static positions of vehicles, the location information is dynamic, following the smartphone's location. Because the volume of signals from smartphones is huge, data preprocessing is needed to generate some indication of traffic jam information.

Currently, traffic prediction has been developed to deal with challenges and concerns as summarized in [8]. With respect to an urban road network, as mentioned in [8], the main challenges that traffic prediction faces include: (1) A more complex road network (large scale) should be considered due to the urban environment, (2) the model should include spatial-temporal (ST) characteristics of the traffic network to describe the corresponding conditions among multiple roads, and (3) utilization of artificial intelligence is still an emerging research area. How to specifically develop a deep learning framework in this area is an important research topic.

Understanding, monitoring, and predicting traffic road conditions, such as the level of traffic jam at certain times, are essential needs of urban transportation [9]. Over the past decade, there have been many studies conducted on the traffic prediction model. Most of the existing models have employed statistical, time series based, probabilistic, and neural network approaches according to the reviews [8,10,11]. The following models (1) time series analysis model, (2) traditional machine learning, and (3) deep learning based model, are summarized below.

#### (1) Time Series Analysis Model

Time series analysis approaches such as ARIMA [12] and sliding window ARIMA (SWARIMA) [9] have been used to study the traffic conditions for the purpose of predictions. Traffic flow data recorded from sensors are frequently noisy, and the short-term traffic prediction is considered as a nonstationary. To deal with this issue, Xie et al. applied a Kalman filter (KL) to handle nonstationary for short-term prediction [13]. In their work, wavelet decomposition analysis was utilized to reduce the noise. Similarly, their model was applied to a relatively simple freeway road network. Yan et al. also proposed a traffic flow prediction based on multivariate time series to understand the traffic patterns on the freeway [14]. In their work on traffic volume, occupancy, and speed, multivariate traffic time series were utilized and converted into a complex network structure.

Most of these approaches mainly focus on the following: (1) predicting traffic flow at a single data point, (2) considering traffic data as a sequence, and (3) finding the patterns of the temporal variation of traffic on one road segment. However, the road traffic condition actually can be propagated among the road network as mentioned in [15]. This means that if one road is badly jammed, the traffic is propagated to other nearby roads which causes the network effect. Because of this propagation characteristic, the road network wide (RNW) traffic prediction which predicts traffic conditions of multiple roads at once is needed, not only based on the time series data analysis, but also the network structure of roads.

#### (2) Traditional Machine Learning Model

In addition to the traditional time series analysis, machine learning models such as support vector machine (SVM), neural network (NN), and deep learning (DL) network have been applied to the traffic prediction domain for decades. Tang et al. proposed a fuzzy neural network model to predict traffic speed by considering periodic characteristics [16]. In their work, k-means was employed to extract periodic features of travel speed data that had been collected from three adjacent stations. Then, a trigonometric regression was used to predict travel speed for multi-step ahead. Although the proposed model performed well on single traffic road prediction, the model did not consider road network perspective, as mentioned before.

Many SVM-based models have been widely employed for traffic prediction. For example, Zeng et al. proposed AOSVR to deal with time efficiency of the traffic flow prediction [17], Saldana-Perez et al. took advantage of social media data to characterize the traffic congestion and analyze crowd-sensed data from a geospatial perspective [18]. Yan, H. and D.-J. Yu. proposed an improved SVM to classify traffic jam conditions which was able to handle a negative effect of an outlier on the traffic data [19]. In their study, relatively small data were used, and therefore their model failed to deal with large-scale traffic prediction.

Furthermore, Tang et al. proposed a hybrid model of SVM with several denoising techniques to predict traffic volume at multiple ahead steps (2 min, 10 min, and 60 min time horizon) [20]. Empirical mode decomposition (EMD), ensemble empirical mode decomposition (EEMD), moving average (MA), Butterworth (BW) filter, and wavelet (WL) were combined with SVM. According to their experiments, obviously, EEMD outperformed as compared with other denoising techniques. Although the proposed model performed well to predict multiple ahead steps traffic conditions, the spatial correlation between the detectors was not considered in their study. Moreover, the predictive model only focused on the freeway and not the urban environment.

### (3) Deep Learning Based Model

Recently, more advanced approaches such as deep learning model have been applied to traffic prediction due to their promising performance. For example, Kim et al. utilized the much deeper and complex recurrent neural network (RNN) model to predict traffic speed [21]. Abbas et al. used long short-term memory (LSTM) model, a type of RNN, for the short-term traffic prediction on the road [22]. In addition, LSTM has also been used for travel time prediction [23]. The RNN and LSTM models are state-of-the-art and powerful for capturing temporal features in traffic. However, the spatial interaction is not considered from the point of view of the road network, although the forecasting task is on a single road in a relatively small region.

Realizing local dependencies of a network can improve the prediction. Song et al.'s work predicted Seoul's main road traffic speed on weekdays utilizing CNN and obtained better results than two multilayered perceptron network [24]. Another research took advantages of both LSTM and CNN to capture a spatial-temporal correlation to predict travel time [25]. Nevertheless, in the literature, few studies have considered the RNW as a whole to exploit the correlation of spatial-temporal features effectively and estimate the traffic interactions among the road segments on a large scale. Especially, for urban roads, knowing the traffic condition in a whole road network, instead of a single road, can result in better decision making by transporters.

To address this concern, Ma et al. proposed a novel approach to learn traffic data as image and applied CNN to predict traffic speed on roads network wide instead of single road segment [26]. The model was evaluated on Beijing's ring road network using traffic data from taxis' GPS across the city and achieved 42% accuracy improvement as compared with other algorithms such as KNN, ANN, random forest, and least square methods. In addition, in studies by [27,28], CNN-based model was also compared to traditional ANN. Their findings confirmed that deep learning based model outperforms the traditional network (ANN), other machine learning methods, and statistical methods.

As evident in the literature, CNN has a drawback in the pooling operation which was addressed in Sabour et al.'s work [29]. To tackle the limitation of CNN, Sabour et al. introduced a new type

of neural network, called capsule network (CapsNet). A capsule is a group of neurons which has different properties from the same entity. It is trained by dynamic routing instead of max-pooling and has a different nonlinearity, namely squash. Since then, many research works applied CapsNet to train the prediction model, including the traffic prediction problems. Extending what Ma et al. has done, Kim et al., applied CapsNet to the traffic speed prediction problem [30]. In their work, the max-pooling operator inside CNN was found to lead to information loss of the interaction among road characteristics in urban transportation. Therefore, replacing the max-pooling operator with routing by agreement algorithm in CapsNet improved the prediction result.

In this study, we develop a prediction model focusing on predicting traffic jam speed on urban roads based on the information collected from citizens' smartphones. In order to differentiate the urban roads where this work focused from the free-flow and well-regulated road transportation, we introduce the term "urban swarming transportation" (UST). Essentially, in UST, the lane marks on most of the UST are not clear or even not existing. It means that all kinds of vehicles, pedestrians, and even animals share the same road. The traffic light system on UST in some developing countries is insufficient or not strictly followed by transporters. Figure 1 shows a typical example of the UST condition (the photo was taken in west Jakarta). As shown in Figure 1, the UST roads are swarmed with all kinds of the mentioned transporters. Note that the road in the picture is not "one-way" and all transporters can use this bidirectional and narrow road.

In order to handle the traffic data collected from mobile phones, data preprocessing is needed to map the traffic speed data originally with longitude and latitude, to the traffic measurement with the road sections. In this study, two deep learning methods, CNN and its extension, CapsNet, are used to train the prediction model to predict the traffic jam speed on the UST roads. Note that the CNN model is the benchmark of our work for comparison purposes. In addition, optimized CapsNet (named OCapsNet) architectures is proposed to change the ReLU nonlinearity at the first two layers of the convolution step. Edgar Squash is applied to the capsule layer in the original CapsNet. We propose the use of some strategies to tweak dynamic routing as mentioned in [31–33], and therefore obtain better prediction performance.



**Figure 1.** An example of urban swarming transportation" (UST) condition (pictures taken by authors on 24 August 2019, in Jl. Tanjung Duren Raya, West Jakarta, Indonesia).

The contributions of this paper are summarized as follows:

- We used traffic data recorded by mobile sensors such as GPS, instead of fixed detectors on the road, as a cost efficiency for traffic prediction on urban roads under UST;

- We proposed CapsNet-based traffic jam prediction as a comparable to CNN-based predictive model to deal with the RNW condition which is the complex road network and spatial-temporal traffic road characteristics under UST;
- We improved the performance of CapsNet by utilizing nonlinearity function in the convolution layer of CapsNet to modify dynamic routing on the two-capsule layer of the original CapsNet.

This paper is organized as follows: Section 2 addresses the techniques used for traffic data preprocessing. Section 3 describes the details of traffic jam speed prediction tools. Section 4 outlines the experimental setup in deep learning methods and the experimental results. Finally, the conclusion and future study directions are mentioned in Section 5.

## 2. Traffic Data Preprocessing

In this research, one year of traffic jam speed data on urban roads in the Jakarta metropolitan has been used as a sample data of UST for studying. The data is collected by the governmental independent smart city division of Jakarta named Jakarta Smart City (JSC). The traffic speed in any Jakarta area is captured in every second interval from Waze mobile app. The measurement of 15 min and 5 min are used to aggregate the traffic jam records. According to information described by JSC, traffic speeds higher than 10 km/h can be considered as free flow (the traffic in Jakarta is extremely congested). Therefore, the collected raw dataset only keeps the traffic jam records. It should be noted that five traffic jam levels are associated with the traffic jam speed (lower than 10 km/h) of each record which comes with longitude and latitude position. Table 1 shows examples of traffic jam speed data. The traffic speed associated with each traffic jam level is categorized as follows:

- Level 0 interpreted as free flow (not recorded);
- Level 1 is 6.1 km/h to 8.1 km/h of traffic speed;
- Level 2 is 4.1 km/h to 6.1 km/h of traffic speed;
- Level 3 is 2.1 km/h to 4.1 km/h of traffic speed;
- Level 4 is bigger than 0.0 km/h to 2.1 km/h of traffic speed;
- Level 5 is 0.0 km/h which is denoted as blocked.

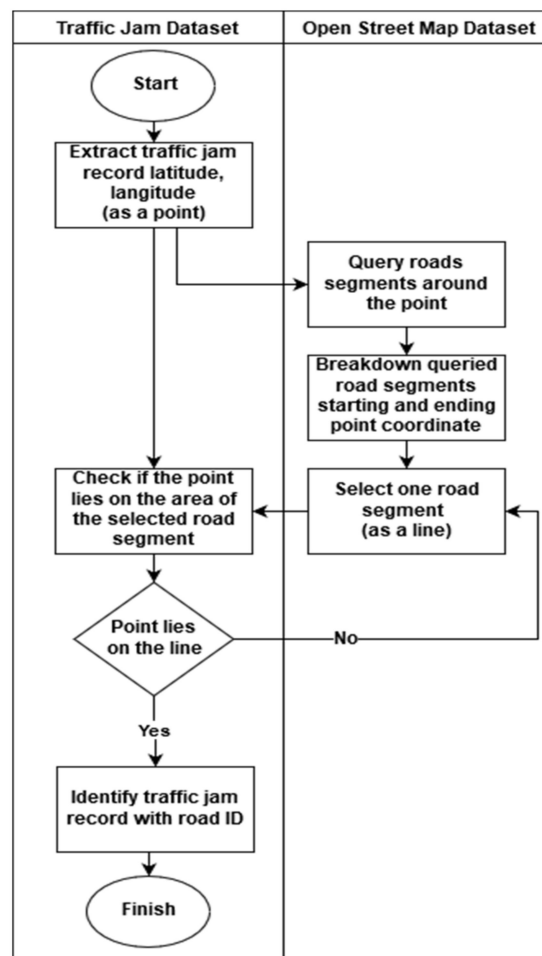
**Table 1.** Examples of traffic jam speed data (Jakarta City).

| - | Longitude  | Latitude  | Speed (km/h) | Level | Time                |
|---|------------|-----------|--------------|-------|---------------------|
| 1 | 106.782318 | -6.198835 | 2.91         | 3     | 2017-11-17 00:18:56 |
| 2 | 106.899734 | -6.218775 | 1.32         | 4     | 2017-11-16 23:34:48 |
| 3 | 106.782875 | -6.333290 | 3.44         | 3     | 2017-11-17 00:04:58 |
| 4 | 106.825172 | -6.187048 | 2.31         | 3     | 2017-11-17 00:02:35 |
| 5 | 106.738623 | -6.126845 | 3.53         | 3     | 2017-11-17 00:11:55 |
| 6 | 106.738625 | -6.126846 | 3.51         | 3     | 2017-11-17 00:11:52 |

Four relevant attributes from the urban traffic jam data were chosen in this research. They are time occurrence, traffic speed, longitude, and latitude of the traffic jam, as shown in Table 1. In order to identify the traffic jam location on a certain urban road, the external road information offered by a public traffic road database (OSM) is used [34].

The process of integrating datasets of traffic jam dataset (Table 1) and OSM is shown in Figure 2. The first step is to extract the coordinate information (latitude and longitude), and traffic jam measure from the dataset as a data point. Secondly, on the OSM, querying out 10 (or more, by setting) road segments which are near to the data point, and connecting the starting and ending coordinates of each found road segment as a line. Third, assuming each road centered with a line has a certain width, check if the point lies on the road. If yes, the traffic measure of the point can be associated with the found road segment. If no, then keep checking if the point can be associated with the other road segment.

If one point does not lie on all found road segments, it means the coordinate of the point is too far from the road and can be considered as a useless point.



**Figure 2.** The integration process of the traffic jam dataset and OSM dataset.

Every road section in the OSM database can be identified with an OSM ID. A single road with a road name may have multiple OSM IDs to represent road sections if the road is very long. Every road section can be broken into smaller road segments identified as Road ID. Adding OSM ID and Road ID, the traffic jam record's coordinates (longitude and latitude) can be located at the same road segment's nearby. Table 2 shows the example of the integrated traffic data with OSM.

**Table 2.** Examples of traffic jam data with OSM data integration (Jakarta City).

| - | OSM ID      | Road ID | Speed (km/h) | Level | Time                |
|---|-------------|---------|--------------|-------|---------------------|
| 1 | 560008540.0 | 9057    | 2.91         | 3     | 2017-11-17 00:18:56 |
| 2 | 28926412.0  | 9295    | 1.32         | 4     | 2017-11-16 23:34:48 |
| 3 | 513190960.0 | 9661    | 3.44         | 3     | 2017-11-17 00:04:58 |
| 4 | 513190965.0 | 102148  | 2.31         | 3     | 2017-11-17 00:02:35 |
| 5 | 459357692.0 | 192345  | 3.53         | 3     | 2017-11-17 00:11:55 |
| 6 | 459357692.0 | 192345  | 3.51         | 3     | 2017-11-17 00:11:52 |

In order to emphasize the research problem, eight main roads in the JSC dataset are chosen, as listed in Table 3. These roads located in the central Jakarta City were used to represent typical Jakarta's urban traffic road with extremely jam-packed traffic conditions. It is noted that multiple kinds of vehicles and pedestrian are allowed to commute on the chosen roads which show a typical

representation of UST traffic condition roads existing in most developing countries. Examples of the road sections (OSM ID) and the associated road segments (Road ID) are shown in Table 4.

**Table 3.** A list of 61 chosen roads for the experiment (Jakarta City).

| - | OSM ID                                    | Road Name                        |
|---|---|----------------------------------|
| 1 | 28809051                                  | S. Parman Rd. (Target)           |
| 2 | 413471399                                 | Other direction of S. Parman Rd. |
| 3 | 566885184, 462712296                      | Tanjung Duren Timur Rd.          |
| 4 | 28225264, 560008540, 594403621            | Tomang Raya Rd.                  |
| 5 | 540317195, 459357692                      | Tanjung Duren Raya Rd.           |
| 6 | 28926412, 566938121, 497546919            | Kyai Tapa Rd.                    |
| 7 | 28931424                                  | Kyai Tapa to S.Parman Turn       |
| 8 | 513190960, 513190965, 297913073, 28926394 | Daan Mogot Rd.                   |

**Table 4.** Examples of OSM ID and the associated Road ID (Jakarta City).

| OSM ID    | Road ID  |
|-----------|--|
| 28809051  | [27390, 56887, 73279, 73313, 85053, 88497, 149973, 153227, 153264] |
| 413471399 | [56883, 153273]  |
| 566885184 | [74322, 74330]   |
| 462712296 | [74328, 115865, 116279, 116282]                                    |
| 28225264  | [27386, 37452, 37460, 152965, 155618, 259258]                      |

In this study, a prediction model was developed to predict traffic flow on the selected road segments which presents typical UST road conditions, as shown in Figure 3, while the information of supplementary road segments is used as additional feed to the prediction. These supplementary roads are chosen by considering target adjacency and representing the main road where spatially correlated with each other. In this study, 2,131,584 rows of fifteen-minute interval data and 6,401,890 rows of five-minute interval data were used. Using these interval data, two datasets are prepared for the experiments. One smaller dataset contained 61 distinct road segments (Road ID) that represented eight distinct selected roads, as shown in Figure 3a. Another larger dataset contained 2972 road segments on  $5 \times 5$  km coverage area, as shown in Figure 3b. Examples of data features used in this work are shown in Table 5.



**Figure 3.** Open street map Jakarta's urban traffic road visualization: (a) Entire Jakarta's urban traffic road map (left) and the chosen Jakarta's urban traffic 61 road segment on 8 main roads (right) and (b) entire Jakarta's urban traffic road map (left) and the chosen more complex Jakarta's urban traffic road  $5 \times 5$  km square area (right).

**Table 5.** Data examples for experiment.

| - | Road ID | Weekday   | Week Num | Time     | Speed (km/h) | Level |
|---|---------|-----------|----------|----------|--------------|-------|
| 1 | 307770  | Wednesday | 52       | 17:45:00 | 10           | 0     |
| 2 | 307770  | Wednesday | 52       | 18:00:00 | 6.340        | 1     |
| 3 | 307770  | Wednesday | 52       | 18:15:00 | 6.347        | 1     |
| 4 | 307770  | Wednesday | 52       | 18:30:00 | 6.856        | 1     |
| 5 | 307770  | Wednesday | 52       | 18:45:00 | 5.616        | 2     |

### 3. Traffic Jam Speed Prediction Model

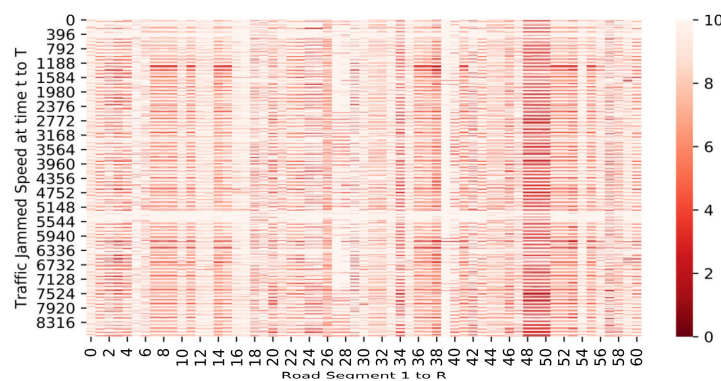
In this study, we aim to propose a traffic jam prediction model that is based on the following considerations: (1) traffic data recorded by mobile sensors such as GPS were used, instead of using fixed detectors on the road, (2) spatial-temporal characteristics of data were used as input image, (3) RNW prediction was conducted to capture the traffic conditions on the entire road network, instead of on a single road, and (4) the CapsNet-based model as an extension of CNN was employed to understand the ST traffic characteristic under UST. In order to achieve the motivation aforementioned above, the traffic jam data are first transformed as an image which contains the traffic jam information of multiple time steps, and feeds into the model. The detailed approaches are introduced in the following section.

#### 3.1. Converting Traffic Jam Speed as an Image

In order to fit the traffic data to the CapsNet network, we then consider the recorded traffic jam speed as a pixel of the one-channel image with R-dimension. It also means a  $T \times R$  matrix should be constructed as denoted in Equation (1), where R is the number of total road segments and T is time horizon representing the number of 5 min intervals. Assuming that  $VS$  is the input space passing to the CapsNet network, the output  $VSOUT$  can be denoted in Equation (2) as an output matrix with  $L \times R$  dimension, where L is the number of time horizon to be predicted. Figure 4 illustrates one example of converting traffic jam data into image.

$$VS = \begin{bmatrix} VS_{11} & \cdots & VS_{1R} \\ \vdots & \ddots & \vdots \\ VS_{T1} & \cdots & VS_{TR} \end{bmatrix}, \quad (1)$$

$$VSOUT = [VOUT_1 \dots VOUT_R, VOUT_{R+1} \dots VOUT_{LR}]. \quad (2)$$



**Figure 4.** The input of the network is considered as the images  $T \times R$ , R is the total road segment, and T is time horizon representing the number of 5 min intervals, and t represents every five min traffic jam speed.



### 3.2. CNN-Based Traffic Jam Prediction

In this research, CapsNet was the proposed method and CNN model was considered as a benchmark to show how the traffic jam speed predictive model as an image perspective can be achieved. To reduce the confusion, we skip the introduction of CNN. Detailed information about CNN can be found in [35]. Here, we only address how the CNN network can be used to deal with traffic jam speed prediction under UST.

Figure 5 depicts the CNN network structure followed by [26] which was used as a benchmark for comparison. Note that the research work in [26] predicted traffic speed in Beijing ring road, but our work not only predicted the traffic jam speeds which are relatively much slower, but also focused on RNW traffic prediction with much more complicated road network topology, in Jakarta. Table 6 shows the parameter settings of the benchmark CNN. In this study, the same parameters adopted in Alex-Net [35] were used, which were the same as in the previous work in [26]. As can be seen, this model is a typical CNN where there are three pairs of convolutions layer with 256, 128, and 64 channels,  $3 \times 3$  kernel, max-pooling of  $2 \times 2$ , and the flattening followed by the fully connected (FC) layer. The stride of two is to capture and learn the traffic jam features, then reshaping to feed the features into the FC layer. The parameters scale of the FC layers depends on the dimension of the given input and the number of historical time step. For example, given the input  $34,944 \times 61$  with 96 historical time steps used, then, the parameter scale of the FC layer is 703,485.

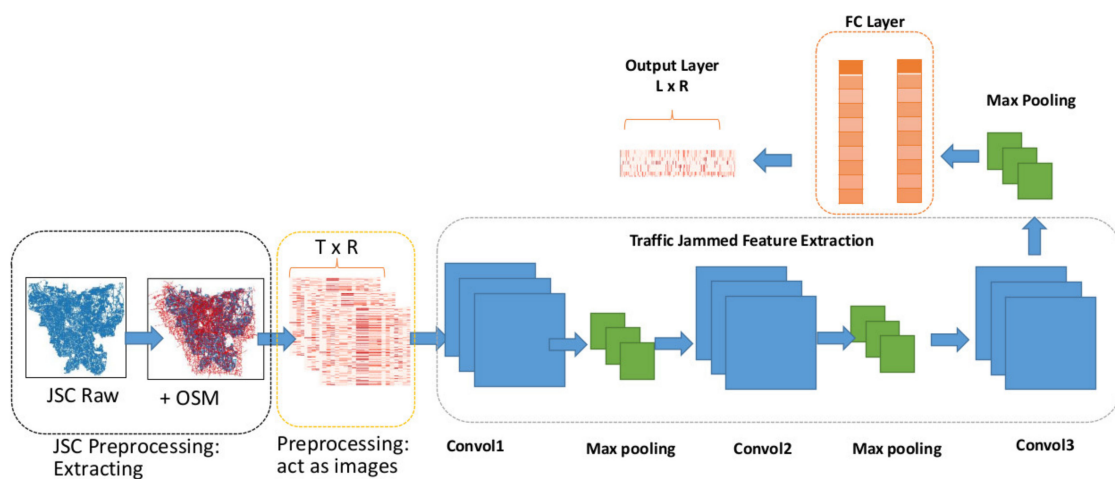


Figure 5. CNN-based traffic jam prediction framework [26].

Table 6. Network settings of CNN based on the work in Ma et al. [26].

| Layer   | Names                | Parameters | Nonlinearity |
|---------|----------------------|------------|--------------|
| Input   | -                    | -          | -            |
| Layer 1 | Convolution-1        | 256, 3, 3  | ReLU         |
| Layer 2 | Max-Pooling-1        | 2, 2       | -            |
| Layer 3 | Convolution-2        | 128, 3, 3  | ReLU         |
| Layer 4 | Max-Pooling-2        | 2, 2       | -            |
| Layer 5 | Convolution-3        | 64, 3, 3   | ReLU         |
| Layer 6 | Max-Pooling-3        | 2, 2       | -            |
| Layer 7 | Flatten              | -          | -            |
| Layer 8 | Fully Connected (FC) | -          | -            |
| Output  | -                    | -          | -            |

### 3.3. CapsNet-Based Traffic Jam Prediction

One drawback of the CNN-based model is that the max-pooling operator only takes the maximum value of the activation. This treatment ignores some of the information about spatial-temporal or the road network. Because all road segments in the urban transportation network are related, a small

change at a certain road segment may affect other road traffic conditions. To address this issue, CapsNet, which uses road network images as inputs, can be applied. CapsNet enables the network to learn and capture the relationship of traffic jam information at certain times and spaces which presents the characteristics of the urban transportation, especially under UST condition.

Unlike CNN, CapsNet works differently where a capsule represents a group of neurons as it activates the neurons by squash nonlinearity. The network is trained using dynamic routing by agreement. The differences of CapsNet and the traditional neural network are summarized as follows:

- Either the input or output of CapsNet is a vector operation where traditional network uses a scalar;
- Before passing to the next layer, CapsNet first transforms its input into its predicted vector so-called affine transformation where traditional network does not have this mechanism;
- After weighted sum multiplies its input, squashing is used to activate the magnitude of the vector, followed by routing to determine which capsule its input should be sent to by coupling coefficient.

There are four layers in the CapsNet model, namely, convolution layer, primary capsule layer, traffic jam capsule, and reconstruction layer. In the convolution layer, a standard convolution and the filter size was applied. Used the standard nonlinearity ReLU written as:

$$\text{ReLU}(s) = \begin{cases} s, & s > 0 \\ 0, & s \leq 0 \end{cases} \quad (3)$$

As for the affine transformation of  $\hat{u}_{ji}$  given in Equation (4), where  $w_{ij}$  is the weight learned in back propagation and  $u_i$  is its input, and the coupling coefficient of  $c_{ij}$  is calculated by the softmax function as denoting in Equation (5). Then the input layer of the parent capsule in the next layer is calculated by Equation (6). The affine transformation was done before the primary capsule layer.

$$\hat{u}_{ji} = w_{ij}u_i, \quad (4)$$

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}, \quad (5)$$

$$s_j = \sum_i c_{ij}\hat{u}_{ji}. \quad (6)$$

The nonlinearity function, so-called squash, given in Equation (7) is carried out to calculate the output vector of the network, Equation (8), where the value is between 0 to 1 which is considered as a probability of the object being present. The more an object is likely to be present the higher the value is yielded, and vice-versa. In the last capsule layer, the loss is calculated for each capsule  $k$ . The loss function is denoted in Equation (9) which is similar to marginal loss function in SVM.

$$\text{Squash} = \frac{\|s_j\|^2}{1 + \|s_j\|^2} \frac{s_j}{\|s_j\|}, \quad (7)$$

$$V_j = \text{Squash}(s_j), \quad (8)$$

$$\text{Loss}_k = T_k \max(0, m^+ - \|V_k\|)^2 + \lambda(1 - T_k) \max(0, \|V_k\| - m^-)^2, \quad (9)$$

where  $T_k = 1$  if and only if a digit classes  $k$  is present,  $m^+ = 0.9$ ,  $m^- = 0.1$ ,  $\lambda = 0.5$  down-weighting of the loss for absent digit classes,  $s_j$  = input matrix multiplication before passing to activation,  $\hat{u}_{ji}$  = prediction vector, and  $c_{ij}$  = coupling coefficient set by dynamic routing iteratively [29].

Equation (10) is the update function of the log probabilities.

$$b_{ij} = b_{ij} + v_j \hat{u}_{ji}. \quad (10)$$

Despite its capability to tackle the CNN limitation, CaspNet suffers severe computational time with respect to more complex data. Therefore, a modification of squash nonlinearity is adopted as in [30]. Thus, the nonlinearity will be more responsive to small shift and the function is given in the following Equation (11).

$$Edgar_{Squash} = 1 - \frac{1}{\exp^{\|S_j\|}} \frac{S_j}{\|S_j\|}. \quad (11)$$

Since the target of our model is traffic jam speed, which is one type of regression problem, then, the loss function is threatened as mean squared error (MSE). Therefore, CaspNet in this research, replacing the loss function by MSE described in Equation (12).

$$MSE(\hat{pv}_k, pv_k) = \sum_{k=1}^R (\hat{pv}_k - pv_k)^2, \quad (12)$$

where  $pv_k$  is the true probabilities associated with an image and  $\hat{pv}_k$  is the calculated probabilities results of the network. In this work, the typical CapsNet setting is shown in Table 7. The performance of this CapsNet is compared with CNN and the optimized version of CapsNet, which are addressed in the following section.

**Table 7.** Network settings of CapsNet.

| Layer   | Name                   | Parameters            | Nonlinearity |
|---------|------------------------|-----------------------|--------------|
| Input   | -                      | -                     | -            |
| Layer 1 | Convol-1               | 32, 3, 3              | ReLU         |
| Layer 2 | Convol-2               | 32, 3, 3<br>(128,3,3) | ReLU         |
| Layer 3 | Primary Capsule        | Capsule: 8            | Squash       |
| Layer 4 | Traffic-Jammed Capsule | Capsule 16            | Squash       |
| Output  | -                      | -                     | -            |

### 3.4. Optimized CapsNet (OCapsNet)

In fact, training CapsNet as described in the previous subsection is time consuming. Xi, E., S. Bing, and Y. Jin, attempted to validate the effectiveness over complex data such as CIFAR10 [29] by increasing the number of primary capsule, stacking more capsule layer, ensembling averaging, modifying the reconstruction loss, and customizing the squash nonlinearity. Their result suggested that ensembling averaging can improve the efficiency, and the number of convolution layers of CapsNet is suggested to be two. Stacking more capsule layer does not influence the result significantly.

Gagana, B., H.U. Athri, and S. Natarajan, investigated the changing of nonlinearity in the convolution layer of CapsNet and tested on MNIST and CIFAR10. Their result shows that Leaky ReLU and variant ReLU, e-swish outperform constantly over the ReLU [32]. Another work by Malmgren [33] surveyed and performed a comparative study of dynamic routing of the CapsNet. Wang and Liu proposed a novel optimization of dynamic routing where the loss function was seen as a clustering-like loss function [31].

On the basis of a literature study, in this work, the optimized CapsNet (called OCapsNet) particularly for traffic jam prediction was proposed. Basically, OCapsNet changes the ReLU nonlinearity as advised in the work by [32] to the leaky ReLU due to its consistency of the performance. The Leaky ReLU function can be denoted as follows:

$$Leaky\ ReLu(s) = \begin{cases} s, & \text{if } s > 0 \\ 0.01s, & \text{otherwise} \end{cases}. \quad (13)$$

The training process in the proposed OCapsNet is illustrated in Figure 6. The format of the input data is tensor (T, R, 1). First, the input data are convolved using conv2D and Leaky ReLU as the activation function into several feature maps  $u_i$ , which is highlighted in blue line, where  $i$  is the number

of feature maps. The feature maps are fed into the primary layer and transformed into the prediction vectors  $\hat{u}_{j|i}$ , where  $j$  is the number of the prediction vectors and  $j|i$  represents the connectivity of the prediction vector  $\hat{u}_{j|i}$  and feature map  $u_i$  through affine transformation, with  $w_{ij}$  as the weights. Then, the prediction vectors  $\hat{u}_{j|i}$  expresses how much the primary capsule  $i$  redound to capsule  $j$ .

Secondly, a product of  $\hat{u}_{j|i}$  and  $c_{ij}$  expresses the agreement between capsules  $i$  and capsule  $j$  is employed to get a single primary capsule  $i$ 's prediction to the class capsule at the traffic jam caps layer where  $c_{ij}$  is the coupling coefficient. The higher the agreement is the more relevant the two capsules are. Thus, increasing the agreement will also be increasing the coupling coefficient.

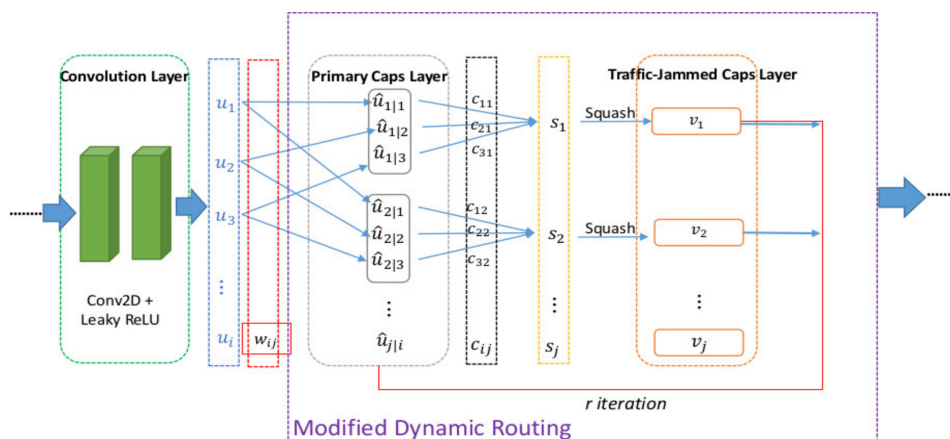


Figure 6. The illustration of training process of the proposed OCapsNet.

Next, a weighted sum  $s_j$  is computed to get the candidates for the squashing function  $v_j$  where the squash to keep the of the output from the capsule is between 0 and 1 as a likelihood. The output from one capsule layer needs to route to the next capsule layer at particular iteration. This routing mechanism is done by dynamic routing (DR), as described in the originally proposed Algorithm 1, and  $c_{ij}$  is updated by finding the dot product  $v_j \hat{u}_{j|i}$  given in the Equation (5).

---

**Algorithm 1** DR (Sabour et al., 2017 [29])

---

1. **Procedure** DR ( $\hat{u}_{j|i}, r, l$ )
  2. **For** caps  $i$  in layer  $l$  and caps  $j$  in layer  $(l + 1)$  :  $b_{ij} \leftarrow 0$
  3. **For**  $r$  iterations **do**
  4. **For** all caps  $i$  in layer  $l$  :  $c_i \leftarrow softmax(b_i)$  (Equation (5))
  5. **For** all caps  $j$  in layer  $(l + 1)$  :  $s_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$  (Equation (6))
  6. **For** all caps  $j$  in layer  $(l + 1)$  :  $v_j \leftarrow Squash(s_j)$  (Equation (7))
  7. **For** all caps  $i$  in layer  $l$  and caps  $j$  in layer  $(l + 1)$  :  $b_{ij} \leftarrow b_{ij} + v_j \hat{u}_{j|i}$  (Equation (10))
  8. **Return**  $v_j$
- 

However, Algorithm 1 has a limitation on the cosine of the angle between two pose vectors is used to measure their agreement. The cosine saturates at 1, which makes it less sensitive to the difference between a quite good agreement and a very good agreement. In fact, the routing procedure can be seen as clustering-like such as the soft k-means algorithm, as discussed in [31]. Therefore, similar approach is carried out in our dynamic routing called modified dynamic routing (MDR) given in Algorithm 2 in this research.

**Algorithm 2** MDR (Similar with [31])

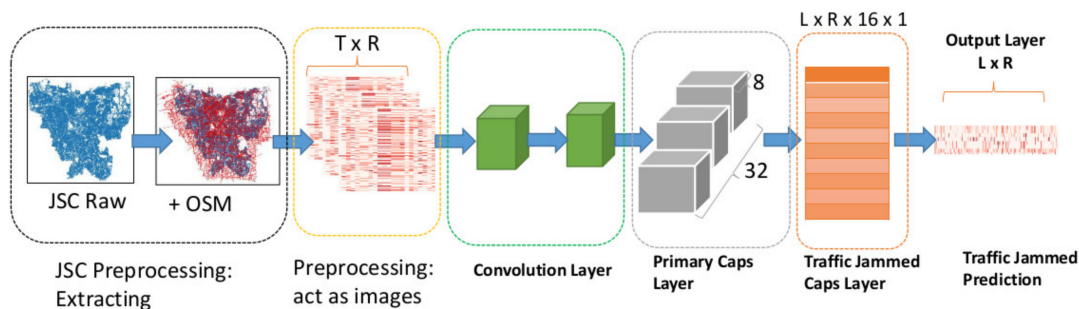
1. **Procedure** MDR ( $\hat{u}_{j|i}, r, l$ )
2. **For**  $r$  iterations **do**
3. **For** all caps  $i$  in layer  $l$  and caps  $j$  in layer  $(l + 1)$ :  $b_{ij} = \frac{1}{\alpha} \langle o_{j|i}, s_j \rangle, c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$
4. **For** all caps  $j$  in layer  $(l + 1)$ :  $\hat{s}_j = \sum c_{ij} o_{j|i}, s_j = \hat{s}_j / \|\hat{s}_j\|$
5. **For** all caps  $i$  in layer  $l$  and caps  $j$  in layer  $(l + 1)$ :  $w_j = \frac{\|\sum c_{ij} o_{j|i}\|}{1 + \max_k \|\sum c_{ik} o_{k|i}\|}$
6. **Return**  $w_j s_j$

The objective function of the MDR algorithm can be defined as follows:

$$\begin{aligned} \text{Min} \left( L(C, S) = - \sum_i \sum_j c_{ij} o_{j|i}, s_j + \alpha \sum_i \sum_j c_{ij} \log c_{ij} \right), \\ \text{s.t. } \sum_j c_{ij} = 1, c_{ij} > 0, \|s_j\| \leq 1 \end{aligned} \tag{14}$$

where  $o_{j|i} = \frac{1}{\|T_{ij}\|_F} T_{ij} \mu_{ij}$  and  $\|T_{ij}\|_F$  is Frobenius norm of  $T_{ij}$ .  $\alpha = 1$ .

Figure 7 illustrates prediction framework of the proposed OCapsNet. Table 8 shows the network settings of OCapsNet.



**Figure 7.** OCapsNet-based traffic jam prediction framework.

**Table 8.** Network settings of the proposed OCapsNet.

| Layer   | Name                | Parameters            | Nonlinearity |
|---------|---------------------|-----------------------|--------------|
| Input   | -                   | -                     | -            |
| Layer 1 | Convol-1            | 256, 1, 1             | Leaky ReLU   |
| Layer 2 | Convol-2            | 32, 3, 3<br>(128,3,3) | Leaky ReLU   |
| Layer 3 | Primary Capsule     | Capsule: 8            | Edgar_Squash |
| Layer 4 | Traffic Jam Capsule | Capsule 16            | Squash       |
| Output  | -                   | -                     | -            |

In short, OCapsNet model can be summarized as follows: (a) In the first two convolution layers use Leaky ReLU rather than standard ReLU, (b) in the primary capsule layer and traffic jam capsule layer apply the modified dynamic routing, (c) investigate Edgar Squash, and (d) finally, perform the prediction at the last layer where using the MSE loss.

## 4. Experimental Results

### 4.1. Experimental Settings

In this study, two datasets were used to evaluate the prediction performance. The first dataset contained traffic jam records from 61 road segments with 15 min interval historical data and the second

dataset contained 5 min interval historical data from 2972 road segment,  $5 \times 5$  km region of West Jakarta. These two datasets represent the simple and the more complex dataset, respectively. Each dataset is divided into 70% for training while 30% for validation and testing. We then normalize the data using minimum and maximum standardization.

In this study, we used TensorFlow deep learning library running under python 3.7. and the computational resources as follows: Processor: i9-9900k 3.6 Ghz, RAM 32 Gb, and GPU NVIDIA@RTX 2080 Ti installed. ADAM optimizer [36] is being used in our implementation with initial learning rate 0.001.

In our experiments, there were four different prediction tasks. These different tasks are conducted to evaluate the performance of the proposed model on different problem complexity. Task 1, Task 2, and Task 3, considered as relatively simple prediction problems in terms of simple road topology, has the prediction tasks on the selected 61 road segments with 15 min interval by considering the previous 96, 24, and 12 observations as the features, respectively. Task 4 represents the more complex problem which has larger input images with 28 road segments in  $5 \times 5$  km square area (larger geographic area) with 5 min interval and 12 previous observations as the features. The tasks are described as the following:

- Task 1, 15 min prediction using 24 hours (t-96) historical traffic jam data on 61 road segments;
- Task 2, 15 min prediction using 6 hours (t-24) historical traffic jam data on 61 road segments;
- Task 3, 15 min prediction using 3 hours (t-12) historical traffic jam data on 61 road segments;
- Task 4, 5 min prediction using hourly (t-12) historical traffic jam speed data on 28 roads segments in a  $5 \times 5$  km square area.

Root mean square error (RSME), mean absolute error (MAE), and mean absolute percentage error (MAPE) are used to evaluate the prediction performance. The smaller RSME is the better result is, as well as MAE and MAPE. Let  $VS_t$  and  $\hat{V}S_t$  be the actual traffic jam speed and the predicted traffic jam speed at time  $t$ , and let  $M$  be the number of total observations, then the performance measurements can be calculated as in Equations (15)–(17).

$$RMSE = \sqrt{\frac{\sum_{t=1}^M (VS_t - \hat{V}S_t)^2}{M}}, \quad (15)$$

$$MAE = \frac{\sum_{t=1}^M |VS_t - \hat{V}S_t|}{M}, \quad (16)$$

$$MAPE = \sum_{t=1}^M \frac{|VS_t - \hat{V}S_t|}{VS_t} \times 100\%. \quad (17)$$

As previously mentioned, there are four prediction tasks in this experiment. In general, each task can be divided into the following two groups according to their prediction area: (1) predictions on the selected 61 road segments and (2) prediction on  $5 \times 5$  km square area. The two groups have a different number of total observations  $M$ , i.e., 34,944 observations in group 1, and 105,120 observations in group 2.

Because different sets of activation function can produce different results [29]. Here, four sets of activation functions were tested on the proposed OCapsNet model to determine which activation should be used. Table 9 shows that evaluation results under one month of traffic data for 28 road segments on the selected  $5 \times 5$  km square area. The best results are indicated in asterisks in Table 9. As can be seen, the Leaky ReLU, Edgar Squash, and Squash are the best for convolution layer, primary capsule layer, and traffic jam capsule layer, respectively. These setting are used for the four prediction tasks and the results are presented in the following section.

**Table 9.** Performance with the different activation function of OCapsNet.

| Convolution  | Primary Caps | Traffic Jam Caps | MAPE   | RMSE  | MAE   |
|--------------|--------------|------------------|--------|-------|-------|
| ReLU         | Squash       | Squash           | 13.351 | 0.448 | 0.519 |
| Leaky ReLU   | Squash       | Squash           | 14.529 | 0.567 | 0.643 |
| * Leaky ReLU | Edgar Squash | Squash           | 12.529 | 0.421 | 0.45  |
| Leaky ReLU   | Edgar Squash | Edgar Squash     | 13.567 | 0.439 | 0.551 |

\* The activation function settings which are then used for the four prediction tasks.

#### 4.2. Experimental Results

Table 10 shows the summarized results on the prediction tasks. Please note that the reported results in Table 10 are the average of MAPE, RMSE, and MAE. The “t-96”, “t-24”, and “t-12” indicate the number of time intervals in the input image. For example, t-96 has 96 historical data points in 15 min intervals. It also means t-12 has fewer historical data points used for training.

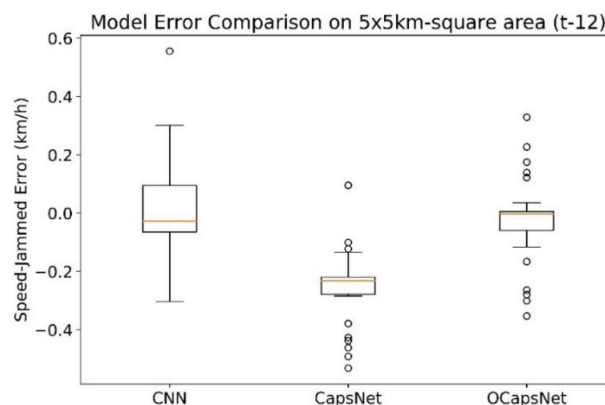
As can be seen, CNN has better result on Task-1 which has more historical data points for training. This result is not surprising because CNN requires more data to maintain the good prediction quality. When the data point in time horizontal is fewer, CapsNet based network outperforms CNN. Particularly, OCapsNet can produce the better results than the original CapsNet.

**Table 10.** Performance comparison on four selected tasks.

| Task             | CNN          |              |              | CapsNet |              |       | OCapsNet     |              |              |
|------------------|--------------|--------------|--------------|---------|--------------|-------|--------------|--------------|--------------|
|                  | MAPE         | RMSE         | MAE          | MAPE    | RMSE         | MAE   | MAPE         | RMSE         | MAE          |
| Task-1 (t-96) *  | <b>29.56</b> | <b>1.707</b> | <b>1.272</b> | 30.46   | 1.749        | 1.307 | 29.74        | 1.709        | 1.274        |
| Task-2 (t-24) *  | 29.42        | 1.702        | 1.282        | 29.28   | 1.701        | 1.266 | <b>29.15</b> | <b>1.670</b> | <b>1.218</b> |
| Task-3 (t-12) *  | 29.84        | 1.748        | 1.296        | 33.72   | 1.673        | 1.473 | <b>29.25</b> | <b>1.603</b> | <b>1.294</b> |
| Task-4 (t-12) ** | 13.99        | 1.166        | 0.592        | 13.15   | <b>1.009</b> | 0.497 | <b>12.52</b> | 1.049        | <b>0.450</b> |

\* Performance comparison on 61 road segments of the selected 8 UST Roads (see Table 3). \*\* Performance on the selected 28 road segments in a 5 × 5 km square (see Figure 3b)

Figure 8 plots all of prediction errors (unit is kilometer/hour) generated by CNN, CapsNet, and OCapsNet. Obviously, CNN has a much wider distribution of errors (standard deviation is 0.17) although the average of error is near zero. On the contrary, the error distributions of CapsNet and OCapsNet are much compact and their standard of the errors are 0.12, and 0.14, respectively. However, the average error of the original CapsNet is lower and shifted from zero. From Figure 8, we can also conclude the proposed OCapsNet has better prediction on traffic jam speed under UST.

**Figure 8.** Boxplot of predicted traffic jam speed errors in km/h for Task-4.

## 5. Conclusions

In this work, a framework for the RNW traffic jam speed prediction under UST condition is proposed using mobile sensor data and a deep learning CapsNet-based network, called OCapsNet. In order to capture the characteristics of the spatial-temporal urban road networks, the data preprocessing on mapping the data point in traffic speed dataset with the OSM was developed. Then, the images representing the road speeds in a certain area were generated in every 5 min interval and 15 min interval. The generating images are the inputs of the deep learning network to predict the traffic jam speeds on the road network at a particular time horizon. In addition, to improve the prediction accuracy, the optimized CapsNet was proposed which performs as follows: (1) replaces the ReLU function by the Leaky ReLU function at first two convolution layers of CapsNet, (2) simplifies the dynamic routing with fewer searching loops, and (3) applies Edgar Squash function as a new activation function in Capsule layer.

The experiments were conducted to evaluate the prediction performance based on the real-world data from JSC. The performances of the original CNN and CapsNet were compared with the proposed OCapsNet. The results show that OCapsNet outperforms CNN (our benchmark) and the original CapsNet in terms of smaller MAPE, RMSE, and MAE when fewer historical data points are fed into the model. The compact error distribution also indicates that OCapsNet has better precision on prediction.

Although the prediction result is promising, the proposed model still has a limitation of suffering severely longer time for training. How to improve the MAPE from 30% to a better result under complex road network of UST instead of on single freeway is still an open question. In addition, the dynamic routing step can be further improved by considering EM routing or other routing mechanism. In addition, it would be interesting to see if the combination of OCapsNet with LSTM can improve the model as it can fully capture the temporal interaction of the traffic road condition. Last but not least, it would be worth considering more features, such as weather, traffic event, and other traffic information represented to the channel of the image to be included for image-basis input of the deep learning model.

**Author Contributions:** Conceptualization, C.-L.Y.; data curation, A.S.C. and H.S.; formal analysis, H.T.; funding acquisition, C.-L.Y. and K.-L.H.; investigation, H.S.; project administration, K.-L.H.; software, A.S.C.; supervision, H.T.; validation, H.T.; visualization, H.T.; writing—original draft, C.-L.Y.; writing—review and editing, C.-L.Y.

**Funding:** This research was funded by the Ministry of National Science and Technology (MOST) of Taiwan to the National Taiwan University of Science and Technology under MOST-106-2221-E-011-106-MY3, MOST-106-2218-E-011-008-MY2, and MOST-108-2218-E-011-026. In addition, this study is financial supported by the “Center for Cyber-Physical System Innovation” from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan. We would also like to thank Wang Jhan-Yang Charitable Trust Fund for their funding support.

**Acknowledgments:** We thank Budhi Wiboro’s help on contacting with Jakarta Smart City for data collection and collaboration. We also appreciate Jakarta Smart City’s data sharing for this research work.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

## References

1. Su, K.; Li, J.; Fu, H. Smart city and the applications. In Proceedings of the 2011 International Conference on Electronics, Communications and Control (ICECC), Ningbo, China, 9–11 September 2011; pp. 1028–1031.
2. Hashem, I.A.T.; Chang, V.; Anuar, N.B.; Adewole, K.; Yaqoob, I.; Gani, A.; Ahmed, E.; Chiroma, H. The role of big data in smart city. *Int. J. Inf. Manag.* **2016**, *36*, 748–758. [[CrossRef](#)]
3. Wan, J.; Liu, J.; Shao, Z.; Vasilakos, A.; Imran, M.; Zhou, K. Mobile crowd sensing for traffic prediction in internet of vehicles. *Sensors* **2016**, *16*, 88. [[CrossRef](#)] [[PubMed](#)]
4. Lv, Y.; Duan, Y.; Kang, W.; Li, Z.; Wang, F.Y. Traffic flow prediction with big data: a deep learning approach. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 865–873. [[CrossRef](#)]



5. Zhao, Z.; Chen, W.; Wu, X.; Chen, P.C.; Liu, J. LSTM network: a deep learning approach for short-term traffic forecast. *IET Intell. Transp. Syst.* **2017**, *11*, 68–75. [[CrossRef](#)]
6. Kim, Y.-J.; Hong, J.-S. Urban traffic flow prediction system using a multifactor pattern recognition model. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2744–2755.
7. Waze. Free Driving Directions, Traffic Reports & GPS Navigation App by Waze. Available online: <https://www.waze.com> (accessed on 4 February 2019).
8. Vlahogianni, E.I.; Karlaftis, M.G.; Golias, J.C. Short-term traffic forecasting: Where we are and where we're going. *Transp. Res. Part C* **2014**, *43*, 3–19. [[CrossRef](#)]
9. Jia, R.; Jiang, P.; Liu, L.; Cui, L.; Shi, Y. Data driven congestion trends prediction of urban transportation. *IEEE Internet Things J.* **2018**, *5*, 581–591. [[CrossRef](#)]
10. Suhas, S.; Kalyan, V.V.; Katti, M.; Prakash, B.A.; Naveena, C. A comprehensive review on traffic prediction for intelligent transport system. In Proceedings of the 2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT), Bangalore, India, 16–17 March 2017; pp. 138–143.
11. Nagy, A.M.; Simon, V. Survey on traffic prediction in smart cities. *Pervasive Mob. Comput.* **2018**, *50*, 148–163. [[CrossRef](#)]
12. Guo, X.; Deng, F. Short-Term Prediction of Intelligent Traffic Flow Based on BP Neural Network and ARIMA Model. In Proceedings of the 2010 International Conference on E-Product E-Service and E-Entertainment, Henan, China, 7–9 November 2010.
13. Xie, Y.; Zhang, Y.; Ye, Z. Short-term traffic volume forecasting using Kalman filter with discrete wavelet decomposition. *Comput. Aided Civ. Infrastruct. Eng.* **2006**, *22*, 326–334. [[CrossRef](#)]
14. Yan, Y.; Zhang, S.; Tang, J.; Wang, X. Understanding characteristics in multivariate traffic flow time series from complex network structure. *Phys. A Stat. Mech. Appl.* **2017**, *477*, 149–160. [[CrossRef](#)]
15. Wang, Z.; Lu, M.; Yuan, X.; Zhang, J.; Van De Wetering, H. Visual traffic jam analysis based on trajectory data. *IEEE Trans. Vis. Comput. Graphics* **2013**, *19*, 2159–2168. [[CrossRef](#)] [[PubMed](#)]
16. Tang, J.; Liu, F.; Zou, Y.; Zhang, W.; Wang, Y. An Improved Fuzzy Neural Network for Traffic Speed Prediction Considering Periodic Characteristic. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 2340–2350. [[CrossRef](#)]
17. Zeng, D.; Xu, J.; Gu, J.; Liu, L.; Xu, G. Short Term Traffic Flow Prediction Based on Online Learning SVR. In Proceedings of the 8th Workshop on Power Electronics and Intelligent Transportation System Guangzhou, China, 2–3 August 2008; pp. 616–620. [[CrossRef](#)]
18. Saldana-Perez, M.; Torres-Ruiz, M.; Moreno-Ibarra, M. Geospatial modeling of road traffic using a semi-supervised regression algorithm (July 2019). *IEEE Access* **2019**. [[CrossRef](#)]
19. Yan, H.; Yu, D.-J. Short-Term traffic condition prediction of urban road network based on improved SVM. In Proceedings of the 2017 International Smart Cities Conference (ISC2), Wuxi, China, 14–17 September 2017; pp. 1–2.
20. Tang, J.; Chen, X.; Hu, Z.; Zong, F.; Han, C.; Li, L. Traffic flow prediction based on combination of support vector machine and data denoising schemes. *Phys. A Stat. Mech. Appl.* **2019**, *534*, 120642. [[CrossRef](#)]
21. Kim, Y.; Wang, P.; Mihaylova, L. Structural Recurrent Neural Network for Traffic Speed Prediction. Proceedings of ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 5207–5211.
22. Abbas, Z.; Al-shishtawy, A.; Girdzijauskas, S.; Vlassov, V. Short-Term Traffic Prediction Using Long Short-Term Memory Neural Networks. In Proceedings of the 2018 IEEE International Congress on Big Data (BigData Congress), San Francisco, CA, USA, 2–7 July 2018; pp. 57–65.
23. Duan, Y.; Lv, Y.; Wang, F.Y. Travel time prediction with LSTM neural network. In Proceedings of the IEEE Conference on Intelligent Transportation Systems, Rio de Janeiro, Brazil, 1–4 November 2016; pp. 1053–1058. [[CrossRef](#)]
24. Song, C.; Lee, H.; Kang, C.; Lee, W.; Kim, Y.B.; Cha, S.W. Traffic speed prediction under weekday using convolutional neural networks concepts. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 1293–1298.
25. Wei, W.; Jia, X.; Liu, Y.; Yu, X. *Travel Time Forecasting with Combination of Spatial-Temporal and Time Shifting Correlation in CNN-LSTM Neural Network*; Springer: Cham, Switzerland, 2018; pp. 297–311.
26. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y. Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. *Sensors (Switzerland)* **2017**, *17*, 818. [[CrossRef](#)] [[PubMed](#)]

27. Liu, Z.; Li, Z.; Wu, K.; Li, M. Urban traffic prediction from mobility data using deep learning. *IEEE Netw.* **2018**, *32*, 40–46. [[CrossRef](#)]
28. Yang, D.; Li, S.; Peng, Z.; Wang, P.; Wang, J.; Yang, H. MF-CNN: Traffic Flow Prediction Using Convolutional Neural Network and Multi-Features Fusion. *IEICE Trans. Inf. Syst.* **2019**, *102*, 1526–1536. [[CrossRef](#)]
29. Sabour, S.; Frosst, N.; Hinton, G.E. Dynamic Routing Between Capsules. In Proceedings of the NIPS'17, 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 3859–3869.
30. Xi, E.; Bing, S.; Jin, Y. Capsule Network Performance on Complex Data. *ArXiv* **2017**, arXiv:1712.03480, 1–7.
31. Kim, Y.; Wang, P.; Zhu, Y.; Mihaylova, L. A Capsule Network for Traffic Speed Prediction in Complex Road Networks. In Proceedings of the 2018 Sensor Data Fusion: Trends, Solutions, Applications (SDF), Bonn, Germany, 9–11 October 2018; pp. 1–6.
32. Gagana, B.; Athri, H.U.; Natarajan, S. Activation Function Optimizations for Capsule Networks. In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19–22 September 2018; pp. 1172–1178.
33. Malmgren, C. *A Comparative Study of Routing Methods in Capsule Networks*; Linköping University: Linköping, Sweden, 2019.
34. OpenStreetMap. OpenStreetMap Indonesia. Available online: <https://openstreetmap.id/en/dki-jakarta/> (accessed on 4 February 2019).
35. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the NIPS'12 25th International Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, USA, 3–6 December 2012; pp. 1097–1105.
36. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, USA, 7–9 May 2015.



© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).