ISSN 2288-4866 (Print)
ISSN 2288-4882 (Online)
http://www.jiisonline.org
J Intell Inform Syst 2014 March: 20(1): 121~131 http://dx.doi.org/10.13088/jiis.2014.20.1.121

Short-Term Prediction of Vehicle Speed on Main City Roads using the *k*-Nearest Neighbor Algorithm

Mohammad Arif Rasyidi
Department of Electrical and Computer Engineering, Pusan National University
(*arifrasyidi@pusan.ac.kr*)

Jeongmin Kim
Department of Electrical and Computer Engineering, Pusan National University
(*jeongminkim.islab@gmail.com*)

Kwang Ryel Ryu
Department of Electrical and Computer Engineering, Pusan National University
(*krryu@pusan.ac.kr*)

Traffic speed is an important measure in transportation. It can be employed for various purposes, including traffic congestion detection, travel time estimation, and road design. Consequently, accurate speed prediction is essential in the development of intelligent transportation systems. In this paper, we present an analysis and speed prediction of a certain road section in Busan, South Korea. In previous works, only historical data of the target link are used for prediction. Here, we extract features from real traffic data by considering the neighboring links. After obtaining the candidate features, linear regression, model tree, and *k*-nearest neighbor (*k*-NN) are employed for both feature selection and speed prediction. The experiment results show that *k*-NN outperforms model tree and linear regression for the given dataset. Compared to the other predictors, *k*-NN significantly reduces the error measures that we use, including mean absolute percentage error (MAPE) and root mean square error (RMSE).

Received : January 28, 2014 Revised : February 28, 2014 Accepted : March 2, 2014
Type of Submission : Outstanding Conference Paper Corresponding Author : Kwang Ryel Ryu

1. Introduction

Traffic speed measures the average speed of all vehicles passing on a particular road in a certain period of time. It is an important measure in transportation as it can be employed in various ways, including traffic congestion detection, travel time estimation, and road design.

Predicting traffic speed accurately is important in the development of intelligent transportation system as it, for instance, can warn users about the potential traffic congestion as well as suggest the fastest route for travel. However, traffic speed is highly influenced by the traffic flow and occupancy of the road which in turn are affected by several factors, for example traffic incidents, weather conditions, time of the day, and day of the week. These features make it difficult

* This research was supported by MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2013-(H0301-13-1012)) supervised by the NIPA (National IT Industry Promotion Agency).

to predict the traffic speed accurately.

In practice, the speed prediction problem can often be seen as time series prediction where we predict the future value by looking at current and historical data. The main idea behind time series prediction is that time series data have two different properties: partially deterministic and partially chaotic (Wu et al., 2004). Future prediction is obtained by reconstructing the deterministic factor from the available data and predicting the random behavior caused by unknown factors.

There are a lot of methods that have been proposed for time series prediction, for example: moving average and Autoregressive Movi ng Average (ARMA) (Rout et al., 2014), exponential smoothing (Billah et al., 2006), linear regression (Nottingham and Cook, 2001), Artificial Neural Network (Kim et al., 2004), and Support Vector Regression (Wu et al., 2004).

With the fluctuate nature of the speed data, choosing the correct prediction method is essential in obtaining good prediction accuracy. In this paper, we use *k*-nearest neighbor algorithm to solve the speed prediction problem.

k-nearest neighbor algorithm (*k*-NN) is one of the oldest and simplest machine learning algorithms. It works by looking for *k* most similar items in the training data to the given query and combine the result to give the desired output. *k*-NN has been widely applied and giving good results in various machine learning problems such as image classification (Boiman et al., 2008), optical character recognition (Matei et al., 2013), and forecasts (Mizrach, 1992; Fernández-Rodrí guez et al., 1999).

There have been several researches on traffic prediction problem. Although not specifically dealing with speed prediction, they provide a general approach on how researchers deal with traffic prediction problem. Sun et al. (2003) use local linear regression model for traffic forecasting. Wu et al. (2004) use Support Vector Regression to solve travel time prediction. Xie et al. (2007) use Kalman filter with discrete wavelet decomposition for forecasting traffic volume.

In most of those researches, only historical data of the target link are used for the prediction. Here we try to improve the prediction by taking the neighboring links data into consideration.

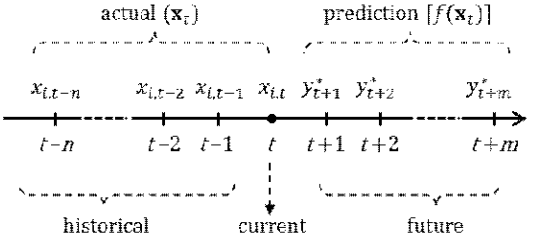
The rest of this paper is organized as follows. The following section provides the formulation of speed prediction problem. In section 3, we describe the *k*-nearest neighbor algorithm. Section 4 gives the experimental procedure and result. Conclusion is given in the final section.

2. Problem Formulation

Let t denote the current time, $x^{*}_{i,j}$ denote the predicted speed for link i at time j, and $x_{i,j}$ denote the actual speed for link i at time j. Given the current and historical speed data $x_{i,j}$ for $i \in all\ links$ and $t - n \leq j \leq t$, we want to predict the future speed for some link k at time $(t+m)$ for some positive values m ($x^{*}_{k,(t+m)}$). m

is called the prediction horizon. It specifies how many steps ahead we want the prediction to be. $(n+1)$ is the window size. It determines how many steps behind we are willing to look back to make our prediction.

Now, let x_j denote the vector containing all current and historical data ($x_{i,j}$), and to simplify the notation let y_j^* and y_j denote the predicted and actual speed for the target link at time *j* respectively as shown in <Figure 1>. Then what we need to do is to find an unknown function *f* which maps x_j to the future value as accurate as possible.



〈Figure 1〉 Speed Prediction Problem

$$\overline{f} = \underset{f}{\operatorname{argmin}} L(\mathbf{y}^*, \mathbf{y}) \tag{1}$$

where *L* is a loss function, $y^*_{j+m} = f(\mathrm{x}_j)$, and ($\mathrm{y}^*$, y) denote the vector of predicted and actual speed respectively.

3. *k*-Nearest Neighbor Algorithm

k-nearest neighbor algorithm (*k*-NN) is a non-parametric method that can be used both for classification and regression. It is a type of instance-based learning or memory-based learning (sometimes is also called lazy learning) where computation is done locally for each query point.

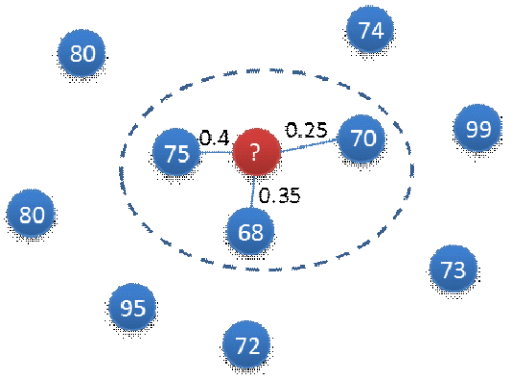
In training phase, *k*-NN simply saves all the training examples and uses them to predict the next example. When given a query, it identifies *k* most similar (nearest) items in the training examples to the query and combines their values to provide the result. Minkowski distance is often used to measure the distance between the examples and the query point:

$$L^p(\mathbf{x}_j, \mathbf{x}_q) = \left(\sum_i \left| x_{j,i} - x_{q,i} \right|^p \right)^{1/p} \tag{2}$$

Let *NN*(*k*, $\mathbf{x}_q$) denote the set of *k* nearest neighbors of query $\mathbf{x}_q$. For classification, we take the plurality vote of the neighbors to assign class to $\mathbf{x}_q$. When doing regression, we can take the mean or median of the neighbors, or solve the linear regression problem of the neighbors (Russel and Norvig, 2010). From this point onward we will talk only about regression case.

A variation of *k*-NN is distance-weighted *k*-NN, first proposed by Dudani (1976). In distance-weighted *k*-NN, each neighbor is given weight depending on its distance to the query point, i.e. closer neighbors are weighted more than the farther ones. An example is shown in <Figure 2>. The weight for *j*-th nearest neighbor is defined as

$$w_j = \begin{cases} \dfrac{d_k - d_j}{d_k - d_1}, & d_k \neq d_1 \\ 1, & d_k = d_1 \end{cases} \tag{3}$$



〈Figure 2〉 An example of weighted *k*-NN with $k = 3$

where d_i denotes the distance of *i*-th nearest neighbor to the query point. The weights are then normalized such that they sum up to 1, i.e.:

$$\sum_{i=1}^{k} w_i = 1 \tag{4}$$

The result for the query is then made by using weighted mean as shown in Equation (5).

$$\overline{y} = \sum_{i=1}^{k} w_i \times y_i \tag{5}$$

One significant drawback of the *k*-NN is their sensitivity to changes in the input parameters, e.g. the number of nearest neighbors, the weighting function, the prediction horizon, and the length of the query vector (Yankov et al., 2006). The best choice of *k* depends on the data. We can use cross validation or experiment with some different values to choose the best *k* for our data. The length of query vector is determined by the features we have. A good set of features will give *k*-NN a good accuracy and vice versa: the performance of *k*-NN can be severely degraded by the presence of noisy or irrelevant features. Feature selection can help with this problem. With feature selection we select a subset of features that allows our algorithm to learn the best model.

4. Experimental Procedure and Result

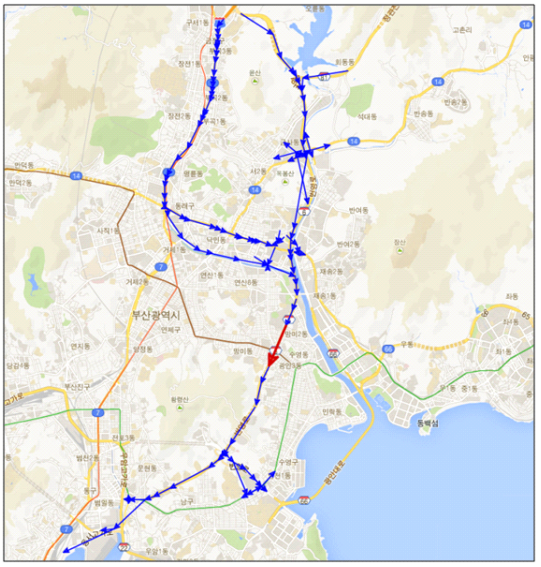
4.1 Dataset

The speed data is provided by Busan Transportation Center. It consists of 6 day-average speed data for all links in Busan road between April 30 and May 5, 2013. The speed data are updated every 5 minutes.

From these data, we select three weekday data for our experiment: from April 30 to May 3 excluding May 1, since it was public holiday. We examine a link that is a part of Beonyeong-ro road in which the speed changes dynamically during the day as shown in <Figure 3>.

〈Figure 3〉 Daily speed in a section of Beonyeong-ro road for April 30, May 2, and May 3, 2013

We experiment with two different kinds of dataset. For the first one, we only use the historical value of the target link. In the second one, we include the speed data of some neighboring links which begin from the toll exit and end at the end of Beonyeong-ro road as shown in <Figure 4>.



〈Figure 4〉 The links chosen for the dataset (Beonyeong-ro). The target link is colored in red, the blue ones are the neighboring links

4.2 Prediction Method

Weighted *k*-NN is used to predict the future speed values. We use Euclidean distance for the distance measure and we experiment with different values of *k*. For performance comparison, we also apply linear regression and M5 model tree to evaluate the performance of *k*-NN.

Linear regression predicts the future value by assuming that the relationship between future value and current and historical data is linear. In other word, future value is represented as dot product of weight vector and historical data as shown in Equation (6). ε_i is called the error term which represents any other factors that influence the target value (y_i) other than x_i. Often this value is defined as a constant.

$$\begin{aligned} y_i &= w_1 x_{i,1} + w_2 x_{i,2} + \cdots + w_n x_{i,n} + \varepsilon_i \\ &= \mathbf{w} \cdot \mathbf{x}_i + \varepsilon_i \end{aligned} \tag{6}$$

M5 model tree divides the data space into smaller subspaces using divide-and-conquer approach (Sattari et al., 2013). It then builds a linear regression model for each subspace. It is one kind of decision tree based regression which uses the concept of local model. This approach differs from regression tree in which for each leaf a constant value instead of linear model is predicted.

4.3 Error Measurement

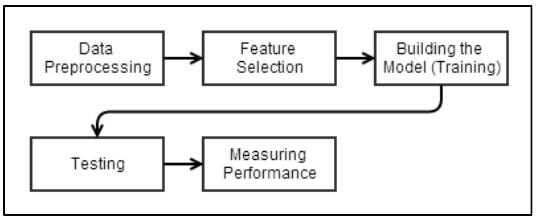
The performance of the predictors is measured using mean absolute percentage error (MAPE) and root-mean-square error (RMSE) measures.

$$\mathrm{MAPE} = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i^* - y_i}{y_i} \right| \tag{7}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i^* - y_i)^2} \tag{8}$$

4.4 Experimental Procedure

We begin by preprocessing the data, followed by performing feature selection, building the model (training), testing the model against the test set, and measuring the performance as shown in <Figure 5>.



〈Figure 5〉 Experimental Procedure

In preprocessing step, the data are converted into six datasets using prediction horizon from one to six (predicting one to six values ahead). The windows size parameter is set to six (we use the speed data from time until). Then we randomly split the dataset into two partitions: 70% of the data are used for training set and the rest 30% are used for testing.

Forward feature selection is used to find the best subset of features for our learning algorithm. We use wrapper method for this purpose. In wrapper method, we evaluate the subset of features using each of the machine learning algorithms that would be employed for learning. Ten-fold cross validation with MAPE as performance measure is used to evaluate the feature subset.

In training step, we train the models for each of the algorithms using the training set that we have prepared before. To find the best model for *k*-NN, we perform ten-fold cross validation to the training data using some different values of *k*: 1, 2, 3, 5, 7, 10, 15, 20, 30, and 50. The same value of *k* is also used in the feature selection step. The resulting models are then applied to the test set. MAPE and RMSE are used to evaluate the performance of each model.

5. Result

The result of *k*-NN experiment with different values of *k* using only historical data and including neighboring links is shown in <Table 1> and <Table 2>. <Table 1> shows that for historical data, *k*-NN tends to obtain best performance when *k* is large. The opposite characteristic is displayed in <Table 2>. When including neighboring links as candidate features, *k*-NN obtains the best performance when *k* is small but greater than one for all prediction horizons, except when the horizon is one. This indicates that historical data cannot really capture the similarities between instances. The result is, when using small *k*, it yields large variance.

In <Table 3>, we show the performance comparison of the models learned using only historical data and the ones learned by including the neighboring links data (All). All methods produce better results when trained by including neighboring links data for all prediction horizons, except for *k*-NN which lost by a small margin on prediction horizon of 1. This suggests that a better set of features can be obtained by inspecting the

〈Table 1〉 Average validation error of *k*-NN using only historical data

MAPE	Prediction Horizon					
k	**1**	**2**	**3**	**4**	**5**	**6**
1	8.98%	13.95%	16.43%	17.38%	19.02%	22.26%
2	7.51%	11.21%	13.03%	14.45%	16.17%	17.81%
3	6.82%	10.63%	12.78%	14.04%	16.02%	17.20%
5	6.64%	9.85%	11.92%	13.37%	15.22%	16.34%
7	6.47%	9.52%	11.62%	12.96%	14.68%	16.12%
10	6.29%	9.55%	11.38%	12.83%	14.47%	15.74%
15	6.12%	9.23%	11.20%	12.63%	14.23%	15.42%
20	6.03%	9.21%	11.14%	12.60%	14.11%	15.42%
30	5.94%	**9.02%**	11.03%	**12.40%**	14.01%	**15.32%**
50	**5.88%**	9.11%	**11.01%**	12.51%	**13.98%**	15.35%

RMSE	Prediction Horizon					
k	**1**	**2**	**3**	**4**	**5**	**6**
1	7.859	11.382	13.391	13.901	15.638	17.252
2	6.709	9.025	10.364	11.282	12.208	13.140
3	6.007	8.536	9.807	10.642	11.732	12.462
5	5.829	7.814	9.230	10.096	11.109	11.751
7	5.715	7.692	9.012	9.856	10.711	11.508
10	5.594	7.768	8.862	9.746	10.665	11.238
15	5.495	7.548	8.735	9.607	10.509	11.111
20	5.486	7.552	8.706	9.641	10.456	11.134
30	5.444	**7.448**	**8.619**	**9.532**	**10.363**	11.060
50	**5.423**	7.492	8.627	9.587	10.384	**10.994**

〈Table 2〉 Average validation error of *k*-NN by including neighboring links data

MAPE	Prediction Horizon					
k	**1**	**2**	**3**	**4**	**5**	**6**
1	9.02%	13.83%	15.76%	17.18%	19.30%	20.72%
2	7.18%	**7.60%**	**7.60%**	**8.06%**	**8.14%**	**8.11%**
3	6.96%	7.76%	8.17%	8.62%	9.10%	8.94%
5	6.57%	8.26%	8.63%	9.24%	9.57%	9.65%
7	6.53%	8.19%	8.80%	9.35%	9.83%	9.97%
10	6.27%	8.42%	8.84%	9.59%	9.85%	10.42%
15	6.15%	8.58%	9.05%	9.70%	10.29%	10.71%
20	6.00%	8.55%	9.22%	9.77%	10.42%	10.99%
30	5.93%	8.50%	9.36%	10.10%	10.70%	11.40%
50	**5.91%**	8.50%	9.75%	10.37%	10.98%	11.72%

RMSE	Prediction Horizon					
k	**1**	**2**	**3**	**4**	**5**	**6**
1	7.882	11.583	12.440	13.597	15.806	16.261
2	6.379	6.574	**6.656**	**6.901**	**6.997**	**7.056**
3	6.118	**6.507**	6.853	7.263	7.443	7.426
5	5.753	6.537	6.970	7.260	7.456	7.588
7	5.838	6.550	7.105	7.297	7.621	7.718
10	5.598	6.642	7.045	7.499	7.654	7.899
15	5.537	6.981	7.098	7.599	7.879	8.140
20	5.459	7.031	7.220	7.610	8.039	8.256
30	5.446	6.994	7.332	7.844	8.169	8.606
50	**5.420**	7.035	7.837	7.946	8.291	8.843

〈Table 3〉 Performance comparison of models trained using only historical data and by including neighboring links data
(LR = Linear Regression, M5 = Model Tree, *k*-NN = *k*-Nearest Neighbor)

k-NN	MAPE		RMSE	
Horizon	**Historical**	**All**	**Historical**	**All**
1	**5.96%**	5.97%	5.541	5.549
2	8.99%	**7.03%**	7.451	**6.178**
3	11.16%	**8.18%**	8.579	**6.782**
4	13.23%	**7.82%**	9.895	**7.023**
5	14.20%	**8.10%**	10.472	**6.861**
6	15.49%	**7.98%**	11.209	**6.834**

M5	MAPE		RMSE	
Horizon	**Historical**	**All**	**Historical**	**All**
1	5.90%	**5.84%**	5.45	**5.263**
2	9.02%	**8.36%**	7.476	**6.95**
3	11.21%	**9.99%**	8.554	**7.762**
4	13.42%	**10.79%**	10.078	**8.368**
5	14.18%	**11.33%**	10.519	**8.401**
6	15.35%	**10.93%**	11.21	**8.386**

LR	MAPE		RMSE	
Horizon	**Historical**	**All**	**Historical**	**All**
1	**5.96%**	**5.96%**	5.591	**5.554**
2	9.28%	**8.86%**	7.675	**7.258**
3	11.59%	**11.07%**	8.887	**8.529**
4	13.84%	**12.60%**	10.529	**9.805**
5	14.76%	**13.21%**	11.092	**9.972**
6	16.11%	**14.47%**	11.827	**10.756**

neighboring links and that the speed of the target link is not only related to its historical values, but also to its neighboring links.

<Table 4> shows the performance comparison of our predictors. While model tree wins by a small margin on the first horizon, *k*-NN clearly dominates on all other horizons. It outperforms the other methods both in terms of MAPE and RMSE. The performance of *k*-NN is very stable across different prediction horizons, giving MAPE between 6% and 8% and RMSE between 5 and 7, while the performance of the other methods deteriorates as the prediction horizon increases.

〈Table 4〉 Performance comparison of the predictors
(LR = Linear Regression, M5 = Model Tree, *k*-NN = *k*-Nearest Neighbor)

Horizon	MAPE			RMSE		
	LR	M5	*k*-NN	LR	M5	*k*-NN
1	5.96%	**5.84%**	5.96%	5.554	**5.263**	5.541
2	8.86%	8.36%	**7.03%**	7.258	6.95	**6.178**
3	11.07%	9.99%	**8.18%**	8.529	7.762	**6.782**
4	12.60%	10.79%	**7.82%**	9.805	8.368	**7.023**
5	13.21%	11.33%	**8.10%**	9.972	8.401	**6.861**
6	14.47%	10.93%	**7.98%**	10.756	8.386	**6.834**

6. Conclusion

From the experiment, we have shown that by including the neighboring links into candidate feature set, we can obtain a better accuracy for our model. Out of the three methods compared, *k*-NN has proven to be superior to both linear regression and model tree. Apart from the first dataset (prediction horizon = 1) in which model tree wins by a little margin, *k*-NN outperforms the other methods in the other prediction horizons, both in terms of MAPE and RMSE. *k*-NN also gives stable accuracy, unlike the other methods whose performance deteriorates as the prediction horizon increases.

In future work, we will experiment with various target links and data of longer period. We will also investigate a formal method in identifying good candidate feature set to further improve the performance of our learning algorithm and accelerate the learning process.

Reference

Billah, B., M. L. King, R. D. Snyder, and A. B. Koehler, "Exponential smoothing model selection for forecasting," *International Journal of Forecasting*, Vol.22, No.2(2006), 239~247.

Boiman, O., E. Shechtman, and M. Irani, "In defense of Nearest-Neighbor based image classification," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2008), 1~8.

Dudani, S. A., "The Distance-Weighted k-Nearest-Neighbor Rule," *IEEE Transactions on System, Man, and Cybernetics*, Vol.6, No.4(1976), 325~327.

Fernández-Rodríguez, F., S. Sosvilla-Rivero, and J. Andrada-Félix, "Exchange-rate forecasts with simultaneous nearest-neighbour methods:

evidence from the EMS," *International Journal of Forecasting*, Vol.15, No.4(1999), 383~392.

Kim, T. Y., K. J. Oh, C. Kim, and J. D. Do, "Artificial neural networks for non-stationary time series," *Neurocomputing*, Vol.61(2004), 439~447.

Matei, O., P. C. Pop, and H. Vălean, "Optical character recognition in real environments using neural networks and k-nearest neighbor," *Applied Intelligence*, Vol.39, No.4 (2013), 739~748.

Mizrach, B., "Multivariate nearest-neighbour forecasts of ems exchange rates," *Journal of Applied Econometrics*, Vol.7(1992), S151~ S163.

Nottingham, Q. J. and D. F. Cook, "Local linear regression for estimating time series data," *Computational Statistics & Data Analysis*, Vol.37, No.2(2001), 209~217.

Rout, M., B. Majhi, R. Majhi, and G. Panda, "Forecasting of currency exchange rates using an adaptive ARMA model with differential evolution based training," *Journal of King Saud University - Computer and Information Sciences*, Vol.26, No.1(2014), 7~18.

Russel, S. J. and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd edition, Pearson Education, 2010.

Sattari, M. T., M. Pal, H. Apaydin, and F. Ozturk, "M5 model tree application in daily river flow forecasting in Sohu Stream, Turkey," *Water Resources*, Vol.40, No.3(2013), 233~ 242.

Sun, H., H. X. Liu, H. Xiao, R. R. He, and B. Ran, "Short term traffic forecasting using the local linear regression model," *Proceedings of the 82nd Annual Meeting of the Transportation Research Board*, Washington, USA, 2003.

Wu, C. H., J. M. Ho, and D.T. Lee, "Travel-Time Prediction With Support Vector Regression," *IEEE Transactions on Intelligent Transportation Systems*, Vol.5, No.4(2004), 276~281.

Xie, Y., Y. Zhang, and Z. Ye, "Short-Term Traffic Volume Forecasting Using Kalman Filter with Discrete Wavelet Decomposition," *Computer-Aided Civil and Infrastructure Engineering*, Vol.22(2007), 326~334.

Yankov, D., D. DeCoste, and E. Keogh, "Ensembles of Nearest Neighbor Forecasts," *Lecture Notes in Computer Science*, Vol.4212(2006), 545~556.

국문요약

*k*Nearest Neighbor 알고리즘을 이용한 도심 내 주요 도로 구간의 교통속도 단기 예측 방법

모하메드 아리프 라시이디* · 김정민* · 류광렬**

교통속도는 교통 문제를 해결하기 위한 중요한 지표 중 하나이다. 이를 이용하여 교통혼잡 탐지, 주행 시간 예측, 도로 설계와 같은 다양한 문제 해결에 활용할 수 있다. 따라서 정확한 교통속도 예측은 지능형 교통 시스템의 개발에 있어 필수적인 요소라고 할 수 있다. 본 논문에서는 대한민국 부산시의 특정 도로를 대상으로 교통 속도에 대한 분석 및 예측을 수행하였다. 과거 연구에서는 대상 도로의 속도 예측을 위해 과거 대상 도로의 교통속도 이력 데이터만을 사용하였다. 그러나 실제 대상 도로의 교통 상황은 인접한 도로의 교통 상황의 영향을 받게 된다. 따라서 본 논문에서는 실제 부산시의 과거 교통속도 이력 데이터를 기반으로 대상 도로와 인접 도로를 모두 고려하여 교통속도 예측 모델의 학습을 위한 속성을 추출하였다. 이와 같이 후보 속성들을 추출 한 후 선형 회귀 (linear regression), 모델 트리 (model tree) 및 *k*-nearest neighbor (*k*-NN) 기법을 이용하여 속성의 부분집합 선택 (feature subset selection)과 교통속도 예측 모델 생성을 수행하였다. 실험 결과 주어진 교통 데이터에서 *k*-NN 기법은 선형 회귀 및 모델 트리 기법에 비해 평균절대백분율오차 (mean absolute percent error, MAPE)와 제곱근평균제곱오차 (root mean squared error, RMSE) 측면에서 더 나은 성능을 보임을 확인하였다.

주제어 : 교통속도 예측, 교통 분석, 속성 추출, 시 계열 데이터, *k*-nearest neighbor

* 부산대학교 전자전기컴퓨터공학과
** 교신저자 : 류광렬
부산대학교 전자전기컴퓨터공학과
2, Busandaehak-ro 63beon-gil, Geumjeong-gu, Busan 609-735, South Korea
Tel: +82-51-510-3531, Fax: +82-51-510-2453, E-mail: krryu@pusan.ac.kr

저 자 소 개

Mohammad Arif Rasyidi

Received a B.S. degree in Information System from the Sepuluh Nopember Institute of Technology in 2012. He is a M.S. Candidate majoring in Electrical and Computer Engineering at the Pusan National University, Korea. His research interests include Machine Learning and Evolutionary Computation.

Jeongmin Kim

Received a B.S. degree in Electronics Engineering and Computer Engineering from the Pusan National University in 2010. He is a Ph.D. Candidate majoring in Electrical and Computer Engineering at the Pusan National University, Korea. His research interests include Machine Learning, Evolutionary Computation, Data Mining and Probabilistic Reasoning.

Kwang Ryel Ryu

Received the B.S. degree and the M.S. degrees in electronics engineering from the Seoul National University, Seoul, Korea, in 1981, and the Ph.D. degree in computer engineering from the University of Michigan, Ann Arbor, in 1992. Since 1993, he has been with Pusan National University, Busan, Korea, where he is currently a Professor at the Department of Electrical and Computer Engineering. His research interests include Machine Learning, Evolutionary Computation, Data Mining and Probabilistic Reasoning.