1-1-2015

A big-data processing framework for uncertainties in transportation data

Jie Yang
University of Wollongong, jiey@uow.edu.au

Jun Ma
University of Wollongong, jma@uow.edu.au

A big-data processing framework for uncertainties in transportation data

Abstract

Transportation infrastructure takes a primary role in urban development planning. To better facilitate or understand the infrastructure status and demands, a huge amount of transportation data such as traffic flow counts has been collected from numerous transportation monitoring systems. Making full use of harvested data samples to discover important patterns has become an increasingly appealing research topic, in which a sophisticated and uncertainty processing framework is required. In this paper, a big-data processing framework is introduced to analyse the transportation data, particularly taking the classification problem of the parking occupation status as an illustrative example. Three modules are implemented to crawl the raw records, generate high-level features, and apply the machine learning algorithm for classification. In addition, the fuzzification algorithm is also introduced to quantify the key attributes of the data, which helps in removing the data redundancy and inconsistency. The proposed framework then is evaluated using a real-world dataset collected from twelve car parks in a university. Simulation results show that the proposed framework performs well with a convincing classification accuracy.

Disciplines

Engineering | Science and Technology Studies

Publication Details

Yang, J. & Ma, J. (2015). A big-data processing framework for uncertainties in transportation data. In A. Yazici, N. R. Pal, U. Kaymak, T. Martin, H. Ishibuchi, C. Lin, J. M. C. Sousa & B. Tutmez (Eds.), IEEE International Conference on Fuzzy Systems (FuzzIEEEE 2015) (pp. 1-6). United States: IEEE.

A big-data processing framework for uncertainties in transportation data

Jie Yang, Jun Ma
SMART Infrastructure Facility, Faculty of Engineering and Information Sciences
University of Wollongong, Northfields Avenue, Wollongong, New South Wales 2522, Australia
E-mails: {jiey, jma}@uow.edu.au

***Abstract*—Transportation infrastructure takes a primary role in urban development planning. To better facilitate or understand the infrastructure status and demands, a huge amount of transportation data such as traffic flow counts has been collected from numerous transportation monitoring systems. Making full use of harvested data samples to discover important patterns has become an increasingly appealing research topic, in which a sophisticated and uncertainty-processing framework is required. In this paper, a big-data processing framework is introduced to analyse the transportation data, particularly taking the classification problem of the parking occupation status as an illustrative example. Three modules are implemented to crawl the raw records, generate high-level features, and apply the machine learning algorithm for classification. In addition, the fuzzification algorithm is also introduced to quantify the key attributes of the data, which helps in removing the data redundancy and inconsistency. The proposed framework then is evaluated using a real-world dataset collected from twelve car parks in a university. Simulation results show that the proposed framework performs well with a convincing classification accuracy.**

I. Introduction

Research indicates that in 2014, approximately 50% population worldwide is living in the urban area [1]. In Australia, this figure is even estimated to be up to 90%. Many challenges in urban development on the way of "Smart City" have been imposed such as sustainable environment, limited resources, and increasing transportation volumes.

Transportation infrastructure including road networks, traffic junctions, as well as car parks is a compulsory component of Smart City. Several studies have demonstrated that transportation infrastructure is one of the main driver towards the urban growth [2–4]. In addition, transportation infrastructure is also the key indicator or reference of urban economic and environmental models [5, 6]. Many research effort has been dedicated to this active area from a variety of perspectives, such as intelligent routing [7], infrastructure sustainability [4], and land use model [8]. Nevertheless, limited study is done to understand the parking occupation model with regarding to other aspects, such as parking location or driver activity. A comprehensive study on the parking occupation has the potential to ease the traffic congestion, contribute to more sustainable mobility.

There are quiet a few possible factors relevant from both the temporal and spatial perspective. For instance, a car park is more likely be full during the peak hours and office time compared to the off hours. Moreover, a car park is more attractive to drivers if it is closer to their destinations. Understanding these possible temporal or spatial factors is vital to modelling the parking occupation, which requires a large variety of data samples. In addition, the amount of the data is expected to grow exponentially. For instance, to have a real-time track of any car park will generate dramatically increasing data. Thus, the study on the parking occupation involves three challenges in terms of the data variety, velocity and volume, which is a typical scenario for the big-data analysis. In addition, uncertainty is ubiquitous in the temporal and spatial scales of a parking model, which is difficult to handle in a traditional way.

In this paper, we present a big-data processing framework, termed *Car Parking Data Miner (CPDM)*, to explore the temporal-spatial effects on the car parking. Three modules are implemented in the CPDM framework, including the data crawler, feature generation and machine learning module. The crawler is used to harvest the raw parking records in a real time. Temporal and spatial features are generated using aggregation function and the fuzzification algorithm at different levels. Finally, the quantified features are trained using the *Support Vector Machine* (SVM) algorithm. All three modules in the CPDM framework are developed based on Apache *Hadoop* and *Spark* platforms.

The CPDM framework is evaluated using a real-world scenario of the University of Wollongong, a primary regional employer in the Wollongong city and a critical node of local traffic network. The main purpose is to investigate the parking pattern associated with different temporal and spatial features. We hope that our preliminary development for analysing quantitative and qualitative big data will offer a template to supplement traditional transportation infrastructure research and discussions.

The remainder of the paper is organized as follows. Section II gives a brief introduction of the SVM training algorithm. Section III presents the CPDM framework, in which three main modules are introduced. Section IV discuss the experiment by applying the CPDM framework to the real-time parking data. Section V presents

concluding remarks.

II. SVM algorithm

In this section, we briefly review the conceptual model for SVM, which is used in the CPDM framework for building parking model. As a supervised learning technique, the SVM algorithm has been demonstrated to perform well in various practical applications [9–13].

An SVM algorithm is fundamentally formulated to address two-class classification problems. The decision boundary is constructed by finding a hyperplane that achieves the maximum separation between two classes. Suppose that we have a training set consisting of N patterns $\{\mathbf{x}_i, z_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^d$ is the i-th input pattern and $z_i \in \{1, -1\}$ is the class label. Assume that the data is linearly separable, the decision function can be written as

$$\begin{cases} \langle \mathbf{w}, \mathbf{x}_i \rangle + b \geq 1 & \text{for } z_i = 1, \\ \langle \mathbf{w}, \mathbf{x}_i \rangle + b \leq -1 & \text{for } z_i = -1, \end{cases} \tag{1}$$

or

$$z_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, \tag{2}$$

where $\mathbf{w}$ is the wight vector, b is a bias term, and $\langle \mathbf{w}, \mathbf{x} \rangle$ is the dot product of the vectors $\mathbf{w}$ and $\mathbf{x}$.

Note that there exist more than one hyperplanes that can separate the two classes; however, only one of them, termed *optimal separating hyperplane*, can maximize the margins between the two classes. Consequently, the SVM training is to find $\mathbf{w}$ and b that achieve the maximum separation:

$$\begin{aligned} &\min \ J(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to } z_i(\langle \mathbf{w}, \mathbf{x} \rangle + b) \geq 1 \text{ for } i = 1, \ldots, N. \end{aligned} \tag{3}$$

The parameter optimization in (3) is built on the assumption of linear separability of the training data. However, if the problem is not linearly separated in the input space, an SVM classifier with a linear decision boundary will have a poor generalization ability. To improve the classification accuracy, the data samples are usually projected from the input space to a higher-dimensional space via a mapping function. As a result, a non-linear decision boundary is constructed for classification.

Overall, SVMs have several important properties including the ability to model complex decision boundaries and reveal hidden relationship between the input and output samples. SVMs have therefore become a very important machine learning tool for many applications.

III. The CPDM framework for car parking analysis

In this section, we first provide background information about the study area. Then the big-data processing framework –CPDM– is presented by introducing three main modules. Employed aggregation functions and fuzzification algorithm are also elaborated herein.

A. Study area

The University of Wollongong is a primary employer with the biggest employee number in Wollongong City Council, New South Wales, Australia. It locates north-west to the Wollongong city and adjoins the free-way link to Sydney. It provides twelve car parks in its main campus with nearly 3100 parking spaces to serve 25000 staff members, students, and visitors daily. With the increasing number of car park users, modelling car park occupation status becomes an important issue for regulating and managing parking facilities in the campus and around area.

B. The CPDM framework

In this section, we present the CPDM framework which effectively investigates the parking pattern using temporal-spatial dynamics. The CPDM framework consists of three main modules (shown in Fig. 1), a brief description of which of them is given as follows:

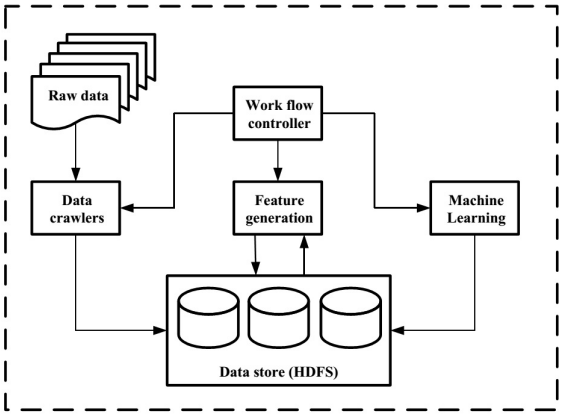

Fig. 1. Architecture of the big-data processing framework of CPDM.

- *Data crawler module:* real-time car-parking data (as records) are collected using a web scraping crawler. Each record is represented as a single-line text and saved into a log file;
- *Feature generation module:* this module helps in identifying significant attributes from the raw data and producing high-level temporal and spatial features;
- *Machine learning module:* SVM algorithm is applied in this module to build up the conceptual model, which associates temporal-spatial features with the parking activity.

A more detailed description of each module and its functionalities is presented in the following.

1) Data crawler module: The parking record is available from the university's publicly-accessible web service (http://www.uow.edu.au/parking/index.html). The content is updated nearly in real time, which enables users to track the real-time availability of parking spaces. Despite of the rich information on the webpage, a systematic procedure to extract the parking status and match them

with temporal or spatial dynamics is not yet provided. In addition, there are other unnecessary content (such as HTML tags or icons) in the webpage to be removed. Therefore, a web scraping crawler is developed to effectively the cope with the target web service while harvesting target records. The following data structure is employed to describe raw records (see Table II).

TABLE I. Data structure for collected raw parking records.

Attribute	Type	Description
Collected Time	Timestamp	Time when the sample is recorded.
Carpark ID	Integer	An unique digit number associated with one specific car park.
Available spaces	String	Empty places in one car park. Whenever there is a special event, this attribute will become a welcome note.

Then, based on the above data structure, the crawler first creates a parse tree based on the HTML tags. Then target contents are extracted by matching the keywords (such as Carpark ID) with the raw HTML tag. Finally, one single car park generates one unique record consisting of three key attributes during each crawling. One record is then represented as a single row of text and stored into a log file. The special symbol "#" is used to separate different attributes. Note that one log file is used to keep records from the same day only.

Since the service enforces some constrains for web requests or data download, the collection interval for the crawler is set to every 2 minutes from 6am to 8pm. This is the time period when the car parks are mainly occupied. Moreover, we are only harvesting the parking data during weekdays.

2) Feature generation module: To interpret the collected raw data – that is, to identify and extract subjective information from the sourced text and turn it into usable data – the feature generation module is implemented to facilitate the data analysis. A feature can be regarded as a user-defined hierarchical representation for the initial content (aka the raw parking records). Features are then used for the classification purpose in the machine learning module.

Again, there are two types of features considered in this paper: temporal and spatial features. These features are produced using the Extraction, Transformation and Load (ETL) process, named herein the aggregation functions, based on different purpose. As academia and students are the main parking users, we designed aggregation functions according to their behaviour and interest. Without loss of generality, three rules are considered to simulate user's requirement in this paper.

- *Semester rule:* during the semester time, more people is expected than the school recess;
- *Slot rule:* different time slots are associated with different parking activity in one day;
- *Region rule:* A car park is more appealing if providing a closer access to users' destination.

The first two rules are used to generate temporal features, while the third one is to describe the spatial condition. Note that more rules can be imported to meet other purposes. Meanwhile, those pre-defined rule should be modified according to different users. Table II further shows proposed aggregation functions exploited for the automatic generation over temporal and spatial features, based on three rules.

TABLE II. Aggregation functions used for generating temporal-spatial features.

Features	Aggregation functions
Temporal	Collected Time → date → Semester Collected Time → time → Slot
Spatial	Carpark ID → distance → Region

The first procedure towards temporal-feature generation involves dividing one raw record into smaller syntactical components using the separator "#". Then the attribute "Collected Time" is selected as the initial input. More precisely, this timestamp attribute is represented as a 18-bit string. The first ten characters contain the "date" information, such as the year, month and day. The second part of the timestamp attribute (i.e., the last eight characters) is for the detail of the "time" context, including the hour and minute data.

Then, the first aggregation function is employed to map the "date" data to a high-level feature "Semester" (or the Semester rule). The conceptual aggregation is conducted according to the semester calendar. If the date falls into the school recess in the semester schedule, the feature is clarified as positive; otherwise, a negative outcome will be generated for a school teaching date. Similarly, to generate the Slot feature (following the Slot rule), the aggregation function is designed by dividing the target time into five slots: 6am-8am, 8am-10am, 10am-3pm, 3pm-5pm, and 5pm-8pm. Among them, the slots from 8am to 10am and from 3pm to 5pm are the peak hours. Usually, a high density of occupation is expected during these slots for the car park. For the slot from 10am until 3pm, there is a relatively stable time. The slot before 8am is for the early birds while the slot after 5pm is used to monitor the status of the parking places in the off hours. As a result, the high-level Slot feature is generated by matching the "time" values to different slots.

As for the spatial feature, the attribute "Carpark ID" from the raw data is used to generate the Region feature (according to the Region rule). The main purpose herein is to measure the activity of the car park against its location. In this paper, the main teaching building is taken into account as the reference point. Therefore, the

distance between the central coordinate of a car park and the reference is calculated.

Furthermore, to discretize the continuous distance value, a fuzzification algorithm is applied which includes the following steps:

- *Step 1:* define several fuzzy sets for a processed attribute;
- *Step 2:* calculate the membership degrees of an observed value with respect to these fuzzy sets;
- *Step 3:* select the fuzzy set with the biggest membership degree as the representative value of an observed value.

To this end, Fig. 2 describes the given fuzzy sets for the "Region" feature. From this figure, it is known that if a car park has the distance over 200 meters is represented as the far Region.

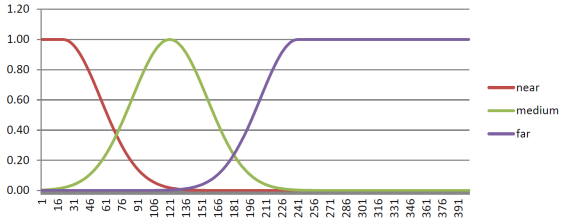

Fig. 2. Example of fuzzification for "distance" (Region).

The fuzzification algorithm is also applied to discrete the current parking status based on the occupation rate:

$$R_{occupation} = 1 - \frac{N_{residual}}{N_{total}}, \tag{4}$$

where $N_{residual}$ and N_{total} is the residual and maximal places for the car park, respectively. Using Eq. (4), the current parking status is normalized to remove the data redundancy and inconsistency, and also for a fair comparison between different sized car parks. Fig. 3 describes the given fuzzy sets for measuring the current status of a car park. A car park is considered as "full", "popular", and "empty" if its $R_{occupation}$ is larger than 70%, between 35% to 70%, and below 35%, respectively.

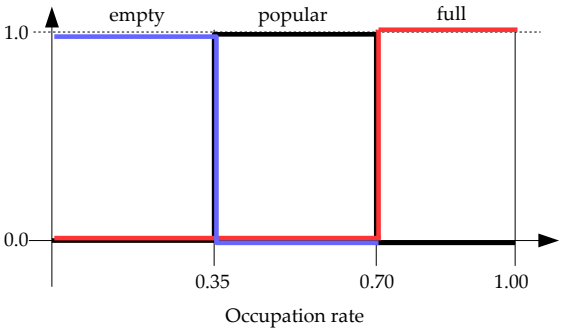

Fig. 3. Example of fuzzification for the parking occupation status.

Using fuzzification technique here can provide a better understanding of the data to meet the perceptions of users and also explain users' behaviour in a natural way.

3) Machine learning module: Although there have been many fundamental advances in the machine learning domain, many problems remain unsolved to date, particularly processing the large number of training samples or big data. For instance, thousands of images or digital records are produced every day. This typically leads to large-scale or high-dimensional data that exceeds the processing capacity of the most sophisticated machine learning algorithms. The varieties and uncertainties of big data samples make it even harder to effectively apply the traditional machine learning algorithm. The Apache Hadoop provides an effective way to handle big data and becomes the industrial standard computing platform.

The Hadoop framework is commonly used to analyse large-scale data sets, such as social network, online web content, or large-scale graph. In the core of Hadoop is the Hadoop Distributed File System (HDFS), which is used to store, distribute and process the large samples. The reliability of HDFS enhances the processing capability for the big data. The Hadoop framework is employed in this paper as the basis of the CPDM framework, whilst the machine learning module is implemented on top of it.

In addition, the machine learning module is implemented using the Spark framework. Spark is an open source environment for fast data analytic. It provides a scalable platform for in-memory computing, thereby achieving advanced performance over other storage approach. MLlib is the machine learning library running under the Spark platform. As cited from its official website, the advantage of MLlib includes: "ease of use and high-quality algorithms (100 times faster than MapReduce)". More importantly, this library is easy to deploy as it supports running on existing Hadoop clusters. As a result, in this paper, the parking data set is stored using the Hadoop framework. Then the MLlib library accesses Hadoop data and performs the in-memory computing. The final outcome is again stored back into HDFS. Fig. 4 illustrates the implemented ecosystem for processing large car-parking data.

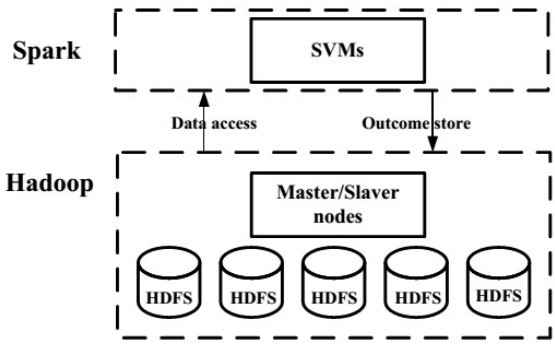

Fig. 4. Implemented ecosystem for processing large car-parking data.

In addition, for multi-class problems, the one-against-one strategy is implemented [14]. That is, $\frac{N_{class} \times (N_{class}-1)}{2}$ classifiers will be constructed if there are N_{class} classes.

IV. Experimental results

This section presents experimental results by classifying the parking status using temporal and spatial features. The employed cloud infrastructure is presented in Section IV-A. The experimental setup of the SVM algorithm and data sets are presented in Section IV-B. The performance of the proposed CPDM framework is then evaluated in Section IV-C.

A. Cloud infrastructure

Virtual cluster is a simple but fast environment to build up the big data ecosystem, i.e., Hadoop and Spark framework. In our implemented cloud infrastructure, a Dell server with Intel Xeon E5-2630 1.8GHz cores and 32G memory is employed. A virtual cluster of four nodes is then deployed. For each node, two virtual CPU and 4GB of memory is allocated. In addition, one node is set up as the master machine for both Hadoop and Spark, while the rest is used as the slaver node. In addition, for the Hadoop platform, the 1.2.1 version is installed. Accordingly, we take Spark 1.1.1 as the running version and the standalone model is adapted to cope with the Hadoop framework.

B. Experimental setup

We harvest the parking data from May to October (2014) in every two minutes from 6am to 8pm using the proposed crawler module. Therefore, for each single day and one car park, the number of records is 30×14=420. Taking twelve car parks into account, as a result, we have 12×30×14×120 = 604800 records in total during the collection period.

The training parameters for the SVM algorithm are set as follows: the maximum number of training iterations is 200; the default L2 regularization with the regularization parameter 0.05 is performed during the iteration; the updating step size is 1.0; and the minimal batch fraction is set as 0.8. The generalization performance is evaluated using the classification error.

C. Performance analysis

In this subsection, the performance of the proposed CPDM framework is evaluated on the classification problem. We mainly consider the effect of the sample size on the training and generalization performance. The percentage of selected samples is computed as

$$m = \frac{M}{Q} \times 100\%, \tag{5}$$

where M is the number of selected samples and Q is the number of all samples (Q =604800). Then, a partitioning into training and test set is given for those selected samples. Moreover, the ratio between the training and test set is always set to 3:2.

Fig. 5 shows the average classification accuracy of the proposed framework with selected samples over 10 runs. When less samples are selected ($m \leq 40\%$), the classification accuracy is lower due to the lack of sufficient information. On the other hand, the performance improves significantly and becomes more stable when more samples are employed. For instance, when the data size increases up to 60%, the generalization performance achieves approximately 82% and approaches the maximal value 83.47%.

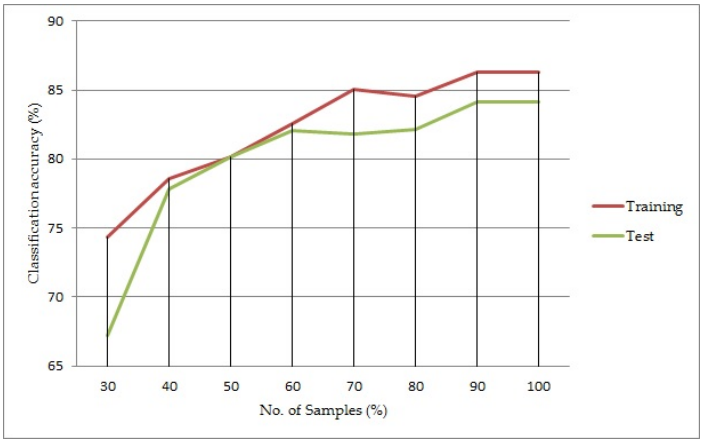

Fig. 5. Average classification accuracy (%) over 10 runs as a function of the size of the selected samples.

Table III shows the computational time of the proposed CPDM framework for different sample sizes. The proposed framework requires slightly more processing time for bigger sample size. However, the difference of average computation time is less than 8.06 second (12%). This demonstrated the stable computational capability of the proposed framework via processing large data sets.

TABLE III. Summary of average computational time (second) for the proposed framework based on different sample sizes.

Data set	m = 30%	m = 40%	m = 50%	m = 60%
Time	58.15	64.12	62.35	63.58
Data set	m = 70%	m = 80%	m = 90%	m = 100%
Time	64.57	63.11	65.28	66.21

Based on the presented results herein, we can conclude that the classification performance of the proposed framework is significantly improved when more data samples are employed. More importantly, the processing time is relatively stable, which indicates that an affordable computation is expected regardless of different sample sizes.

V. Conclusion

In this paper, we proposed a complex framework for big-data processing. Three modules are introduced to crawl the raw online records, generate key features to represent original samples, and then conduct the machine learning algorithm to further classify patterns.

The proposed framework is implemented using two cutting-edge platforms: Hadoop and Spark. The Hadoop platform is used as the fundamental to store the harvested large data sets; while the Spark platform is employed to build up the classifier, in which a multiple-

class Support Vector Machine algorithm is implemented. The proposed framework is then employed on the study of the car-parking model. Temporal-spatial features are generated, using aggregation functions and fuzzification algorithm, which are later used to classify the parking status. Experimentally, the proposed framework leads to a convincing generalization capability.

We hope that our findings will offer an expanded method of analysing transport infrastructure, while also demonstrating the potential for further development in this area. Our next step will be to drill deeper into the dataset in order to observe significant shifts.

References

[1] H. Jung, H. Marguerite, and C. Hu, "Towards an effective framework for building smart cities: Lessons from Seoul and San Francisco," *Technological Forecasting and Social Change*, vol. 89, pp. 80 – 99, 2014.

[2] V. Stefan, "Achieving satisfaction when implementing transportation infrastructure projects: a qualitative comparative analysis of the highway project," *International Journal of Project Management*, vol. 33, no. 1, pp. 189 – 200, 2015.

[3] R. A. Bismark, "An empirical analysis of three econometric frameworks for evaluating economic impacts of transportation infrastructure expenditures across countries," *Transport Policy*, vol. 35, pp. 304 – 310, 2014.

[4] A. Assa, N. Abdul, T. Roshana, and L. Siti, "Transportation infrastructure project sustainability factors and performance," *Procedia - Social and Behavioral Sciences*, vol. 153, pp. 90 – 98, 2014.

[5] M.-D. Lucia, P. Antonio, and M. V. Jose, "Transportation infrastructure impacts on firm location: the effect of a new metro line in the suburbs of madrid," *Journal of Transport Geography*, vol. 22, pp. 236 – 250, 2012.

[6] T. Tong, T. Yu, S.-H. Cho, K. Jensen, and D. Ugarte, "Evaluating the spatial spillover effects of transportation infrastructure on agricultural output across the United States," *Journal of Transport Geography*, vol. 30, pp. 47 – 55, 2013.

[7] T. Eiichi and S. Hiroshi, "Intelligent transportation system based dynamic vehicle routing and scheduling with variable travel times," *Transportation Research Part C: Emerging Technologies*, vol. 12, no. 34, pp. 235 – 250, 2004.

[8] Y. Ahmet, a. Volkan, and D. Hlya, "An evaluation framework for land readjustment practices," *Land Use Policy*, vol. 44, pp. 153 – 168, 2015.

[9] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[10] T. Joachims, "Making large-scale Support Vector Machine learning practical," in *Advances in kernel methods: support vector learning*, pp. 169–184, 1998.

[11] M. Fernández-Delgado, E. Cernadas, S. Barro, and D. Amorim, "Do we need hundreds of classifiers to solve real world classification problems?" *Journal of Machine Learning Research*, vol. 15, pp. 3133–3181, 2014.

[12] J. Lu, B. Liu, G. Zhang, Z. Hao, and Y. Xiao, "A situation assessment approach using Support Vector Machines as a learning tool," *International Journal of Nuclear Knowledge Management*, vol. 3, pp. 82–97, 2008.

[13] J. Lu, X. Yang, and G. Zhang, "Support Vector Machine-based multi-source multi-attribute information integration for situation assessment," *Expert Systems with Applications*, vol. 34, pp. 1333–1340, 2008.

[14] J. Milgram, M. Cheriet, and S. Robert, "One against one or one against all: Which one is better for handwriting recognition with SVMs," *Tenth International Workshop on Frontiers in Handwriting Recognition*, 2006.