# A Convolutional Neural Network for Traffic Information Sensing from Social Media Text*

Yuanyuan Chen, Yisheng Lv, Xiao Wang and Fei-Yue Wang

*Abstract*— **Mining social media data to obtain traffic relevant information is an emerging topic due to the real-time and ubiquitous features of social media. In this paper, we focus on a specific issue in social media mining that concerns to extract traffic relevant microblogs from the Sina Weibo platform, which is the first and essential step to further extract detailed traffic information, such as the location of a traffic incident. It is transformed into a machine learning problem of short text classification. We employ deep neural networks to classify microblogs into traffic relevant and traffic irrelevant ones. More specifically, we firstly adopt the continuous bag-of-word (CBOW) model to learn word embedding representations based on the dataset of three billion unlabeled microblogs. Next we use a convolutional neural network (CNN) to learn the abstract features of traffic relevant and traffic irrelevant microblogs. The key advances in this paper are: use of semantics of words and deployment of deep neural networks to extract traffic information from social media text. Experiments show that the proposed deep learning method has superior performance over support vector machine (SVM) based method and multi-layer perceptron (MLP) based method.**

## I. Introduction

Traffic information, such as traffic incidents and real-time traffic status, plays a fundamental role in improving the efficiency of Intelligent Transportation Systems (ITS) [1]–[3]. Conventionally, traffic information is obtained from physical sensors like GPS, cameras, or loop detectors. Recently Social media has also been regarded as the potential source to serve as social sensors to extract traffic information, since people and authoritative agencies often post transportation information online with the popularity of such platforms. As these platforms have a great number of real-time user-generated contents, they have become powerful and inexpensive information sources [4]–[9].

Mining social media data to extract information has gained a lot of attention in a variety of topics, such as natural disaster detection, epidemics monitoring and crisis response and management. Sakaki et al. analyzed the real-time nature of Twitter, and used Twitter as a social sensor for real-time detection of earthquakes with high probability [10]. Middleton et al. proposed a real-time crisis mapping platform matching location data for areas at risk of natural disasters to geoparsed tweet contents [11]. Aramaki et al. used the support vector machine method to extract influenza epidemics from tweets and could detect influenza epidemics with high correlation [12].

In the field of transportation, social media based traffic research papers mainly focus on traffic event detection and traffic prediction. Researchers used the features of traffic-related keywords in tweets or microblogs to detect traffic events. D'Andrea et al. developed a system based on text mining and machine learning algorithms for real-time traffic event detection from Twitter stream analysis. Experiments show that the system can detect traffic events almost in real time and often before online traffic news websites and local newspapers [13]. Gu et al. proposed a methodology to mine tweet texts to extract incident information on both highways and arterials, and they applied the methodology in the Pittsburgh and Philadelphia Metropolitan Areas in September 2014 [14]. Cui et al. developed a prototype system that extracts traffic accidents and traffic statuses from Sina Weibo using natural language processing and machine learning methods. They also published these traffic events through an Android based application [15]. Gutierrez et al. used tweet messages from regional traffic agencies in UK to detect traffic related events and geo locate the events on a map to notify promptly users [16]. Tejaswin et al. extracted traffic related entities from tweets based on background knowledge from structured data repositories, and then used this data for incident clustering and prediction [17]. Kurkcu et al. captured traffic incident information from web-based map providers and Twitter data, and further incorporated incident information into their proposed virtual sensor methodology for travel time collection [18]. Zhang et al. combined latent Dirichlet allocation and document clustering models for semantic filtering of incident-topic tweets [19].

The rich information embedded in online social media data can help improve traffic prediction. He et al. developed a linear regression model incorporating traffic data and Twitter data to predict longer-term traffic flow where the forecasting horizon is beyond 1 hour [20]. Experiments show that the proposed model outperforms the existing auto-regression based traffic flow prediction model. Ni et al. used social

*This work was partially supported by National Natural Science Foundation of China (61533019, 71232006, 61233001, 71472174, 61603381) and the Early Career Development Award of SKLMCCS.

Yuanyuan Chen is with State Key Laboratory of Management and Control for Complex Systems Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China. He is also affiliated with University of Chinese Academy of Sciences Beijing, 100190, China. (e-mail: yychen5133@ia.ac.cn)

Yisheng Lv is with State Key Laboratory of Management and Control for Complex Systems Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China. (e-mail: yisheng.lv@ia.ac.cn). Corresponding author of this paper.

Xiao Wang and Fei-Yue Wang are with State Key Laboratory of Management and Control for Complex Systems Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China. They are also affiliated with Qingdao Academy of Intelligent Industries Qingdao, Shandong, 266109, China. (e-mail: x.wang@ia.ac.cn, feiyue@ieee.org)

978-1-5386-1526-3/17/$31.00 ©2017 IEEE

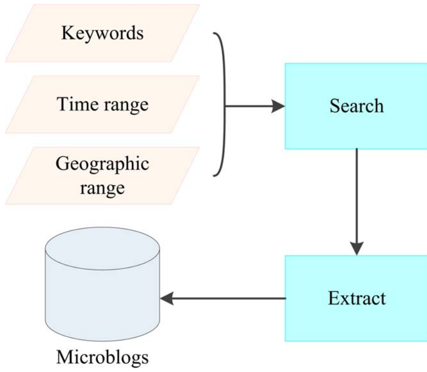


Fig. 1. Framework of Data Collection

media data to develop a short-term traffic flow prediction model under sport game events [21]. Tweet rate features and semantic features are incorporated the prediction model. Experiments demonstrate that including tweet features can improve traffic flow prediction performances. Grosenick extracted non-recurring events like traffic accidents from twitter data and incorporated this information to predict traffic speed on a single road segment with artificial neural networks [22]. Abidin et al. developed a Kalman filter model incorporating traffic information from social networks, to predict bus arrival time [23], [24]. Ni et al. extracted event information from social media data, and used both historical transit data and real-time social media data to predict subway passenger flow under event occurrences [25].

Previous researches have introduced the text mining techniques to traffic information extraction and analysis. However these methods are based on small data and fail to utilize the semantics of sentences or documents. In this paper, we firstly learn word embedding representations based on three billion microblogs. The word embeddings could capture the semantics of words and reduce the dimensions of microblog feature spaces. Then we train a convolutional neural network to extract traffic relevant microblogs form Sina Weibo. Our experiments show that the proposed method can extract traffic relevant microblogs with the highest F measure using less memory compared to competitive methods.

The remainder of this paper is organized as follows. Section II presents data acquisition and prepossessing techniques used in this paper. Section III describes the methodology including word embedding models and convolutional neural network for microblog classification. Section IV presents experimental results. Section IV gives conclusions and suggestions for future work.

## II. Data Acquisition and Preprocessing

### A. Data Acquisition by Crawling Webpages

In this paper, we collect microblogs from Sina Weibo platform by crawling its webpages as illustrated in Fig. 1. First, we send a search request with keywords, time range, and geographic range to Sina Weibo and download the response source codes as HTML files. Then, regular expressions are utilized to match the microblog content and corresponding attribute such as the microblog ID, user ID and timestamp. Last, we store the structured data in a database for future retrieval.

### B. Word Segmentation and Indexing

There is no word delimiter in Chinese sentences compared to English sentences. Thus the first step to Chinese natural language processing is segmentation. And the performances of word segmentation are critical for the following text mining. Among the several source tools to segment Chinese words, we choose ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System) in this paper, which is widely used and has high-performance [26].

After Chinese word segmentation, learning word embeddings and building microblog feature vectors could be done directly. However as the size of vocabulary is large, the direct searching method is inefficient. In this paper, we build a search tree for efficiently accessing the vocabulary. And the Huffman tree is selected as the search tree. It is worth to mention that the word2vec model given in the next section also needs to construct a Huffman tree. Thus we build a Huffman tree once for both training word embedding representations and constructing microblog feature vectors. The process to construct a Huffman tree is illustrated in Fig. 2. By initializing an empty vocabulary, the first step is to read segmented microblogs, and then it needs to decide whether there are new words for the vocabulary. The next step is to add the new words to the vocabulary firstly if there are any ones. Then update the number of occurrences of words. The last step is to build the Huffman tree based on the frequency of words, in which words with higher frequency end up with shorter root-to-leaf paths.

## III. Methodology

### A. Word Embedding

Natural language processing systems traditionally represent each word as a one-hot vector [27]. It is a vector filled with 0s, except with 1 at the position associated with the word. The one-hot representation is very high-dimensional and sparse. Moreover, such representation cannot capture

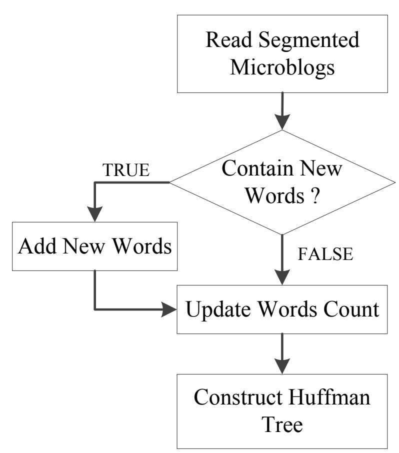


Fig. 2. Flowchart of Data Prepossessing

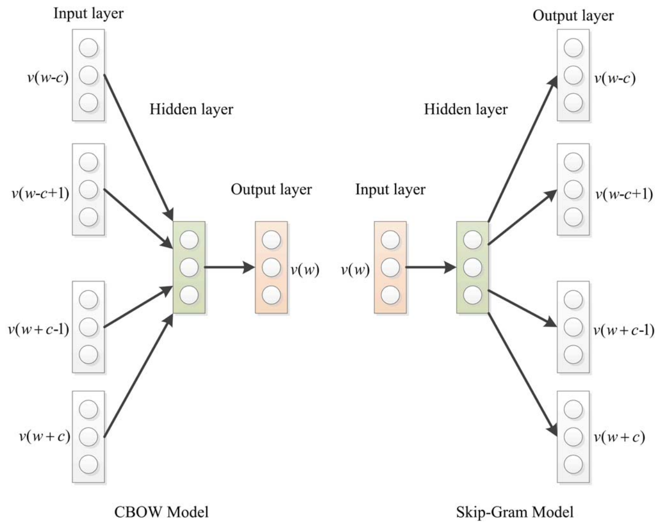


Fig. 3. Model Architecture of CBOW and Skip-Gram

semantic similarity between words. Representing words in a continuous vector space has been proved to be effective in natural language processing tasks by grouping similar words. Recently, using neural networks to get the word representations are very attractive and interesting because the learned vectors explicitly encode many linguistic regularities and patterns. The Continuous Bag-of-Word (CBOW) model and the Skip-Gram model are two well performed approaches to learn word embedding representations [28]–[31]. These two methods mirror each other. The objective to be optimized of the CBOW model is to find word vector representations that are useful for predicting the middle word under a context, while the Skip-Gram model tries to learn word vector representations by maximizing the probability of predicting surrounding words based on the middle word as shown in Fig. 3. In this section, we focus on the derivation of the CBOW model. Meanwhile due to high computational complexity of the original CBOW model and the Skip-Gram model, hierarchical softmax or (and) negative sampling are applied in the training processes. Hierarchical softmax uses a Huffman tree, a binary tree, to code all of the words in the vocabulary. In the CBOW model with hierarchical softmax, the hidden layer is designed to average the input word vectors, so that the output of hidden layer is:

$$h = \frac{1}{C} \sum_{u \in context(w)} v(u) \tag{1}$$

where $v(u)$ represents the vector of the word $u$, $context(w)$ is the set of contextual words of the word , and $C$ is the cardinality of the set . Given the context, the conditional probability of the word $w$ is:

$$\begin{aligned} p(w|context(w)) = \prod_{j=1}^{L(w)-1} & \Big\{ \sigma(h^{\mathrm{T}} v'_{n_{w,j}})^{(1-d^w_{j+1})} \\ & * \Big[ 1 - \sigma(h^{\mathrm{T}} v'_{n_{w,j}})^{(1-d^w_{j+1})} \Big] \Big\} \end{aligned} \tag{2}$$

where $n_{w,j}$ is the $j-th$ inner point from the root to word $w$ in the Huffman tree, $v'_n$ is the vector of inner point $n$, $L(w)$ is the length of the path in Huffman tree for word , and $d^w_{j+1}$ is the $j-th$ bit of Huffman code for word $w$. It is straightforward to train the neural network by maximizing the conditional probability in (2) for a target word . Take the logarithm of the conditional probability and define the loss function:

$$\begin{aligned} L &= \log p(w|context(w)) \\ &= \sum_{j=1}^{L(w)-1} \Big\{ (1-d^w_{j+1}) \log \sigma(h^{\mathrm{T}} v'_{n(w,j)}) \\ &\qquad + d^w_{j+1} \log \Big[ 1 - \sigma(h^{\mathrm{T}} v'_{n(w,j)}) \Big] \Big\} \end{aligned} \tag{3}$$

By maximizing the loss function using stochastic gradient descent method, we can learn the parameters in an iterative way.

### *B. CNN for Text Classification*

CNN is a feedforward neural network that consists of convolutional layers interspersed with pooling layers as shown in Fig. 4 [32], [33]. A convolutional layer aims to learn the region features. The convolution operation can be summarized as moving a filter over the sentence matrix (input map) and computing the dot products as shown in Fig. 5. Convolution with one filter outputs a feature vector. To learn more sophistical features, there are generally a few distinct filters to convolve the input map, and all feature vectors are concatenated into a new matrix called feature map that would be passed to a pooling layer [34]–[36].

Given a sentence $s = w_1 w_w \cdots w_N$ consisting of $N$ words, among which word $w$ can be represented by $V-dimensional$ word vector, sentence $s$ can be transformed into a $N \times V$ dimensional matrix $S$ called the input map. The filter is a $M \times V$ dimensional matrix $K$. Considering one filter $p$ only, the $i-th$ element of the feature vector is:

$$a^p_i = f(conv([v(w_i) \cdots v(w_{i+M-1}), K^p]) + b^p) \tag{4}$$

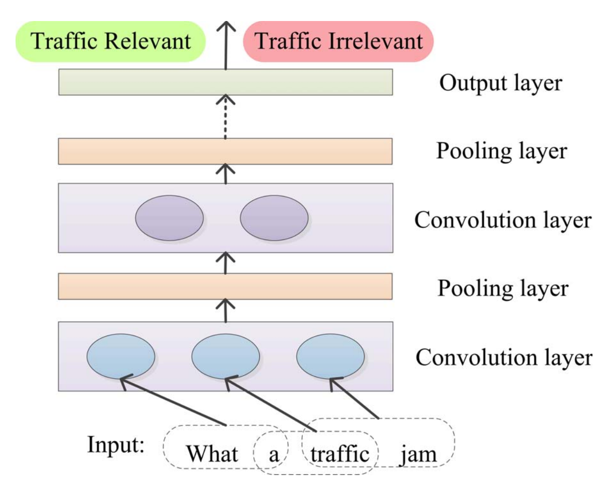


Fig. 4. CNN Architecture for Text Classification

where $K^p$ is filter matrix, $b^p$ is the bias and $f$ is the activation function. And the operation $conv(\cdot)$ is defined as:

$$conv([v(w_i)\cdots v(w_{i+M-1}), K^p]) = \sum_{r=1}^{M} K^p_{(r,:)} v^{\mathrm{T}}(w_{i+r-1}) \tag{5}$$

Substituting (5) into (4), we obtain:

$$a_i^p = f(\sum_{r=1}^{M} K^p_{(r,:)} v^{\mathrm{T}}(w_{i+r-1})) \tag{6}$$

Applying (6) repeatedly from equals 1 to $(N - M + 1)$, then we have the full feature vector $A^p = (a_1^p, a_2^p, \cdots, a_{N-M+1}^p)^{\mathrm{T}}$ of the convolution layer under the filter $p$. If there are $F$ filters in this case, the dimension of output map is $(N - M + 1) \times F$.

The pooling operation computes the average or maximum of each region on the column of the feature map. This operation reduces the dimension of feature map by merging neighboring points, so that computation time is reduced and more abstract information would be passed to the next layers. By minimizing the categorical crossentropy $L$ , we can obtain the optimized parameters, which are here denoted as $\theta$. More formally,

$$\theta = \arg\min_{\theta} \quad L(\bar{Y}, Y) \tag{7}$$

where $\bar{Y}$ is the output distribution of a CNN model and $Y$ is the true value distribution.

## IV. Experiments

### A. Dataset Description

There are two datasets, one dataset is collected for learning word embedding representations and another one is used to train classification modes.

AS learning word embedding is transformed into an unsupervised machine learning issue, it is probable that we construct a large dataset. We have collected about 3 billion microblogs so far. And this dataset covers one million users. All these microblogs are utilized to build a Huffman tree as described in section II, which would be used to train word embedding model and transform the labeled microblogs to feature vectors.

The second dataset is used to train classification models. This dataset is manually labeled as training models to classify microblogs into traffic relevant and traffic irrelevant ones is a supervised machine learning problem. The microblogs, posted from January 2010 to December 2015 in Shanghai, China, were collected by searching Sina Weibo with key words that are manually chosen. The key words adopted are given in Table I, and we have collected 40 thousand microblogs. From the 40 thousand candidate microblogs, we manually selected and labeled 5 thousand microblogs as traffic relevant ones and other 6 thousand microblogs as traffic irrelevant ones.

The labeled microblogs were firstly segmented with ICTCLAS. Then they are transformed into a sequence of word embedding representations by searching the Huffman Tree. To feed into deep neural network efficiently, the word sequence of each microblog for training the classifier was padded into a sequence with the same length. In our experiment, the maximum length of word sequences in manually labeled dataset is 156. And zero vectors were padded after each sequence until its length reached 156. As the dimension of word vector is 200 (more details in next subsection), the features of a microblog fill a $156 \times 200$ matrix, which would be feed into classification models. All the models employed in this paper would utilize this feature matrix as input if there is no additional declaration.

TABLE I
KEYWORDS TO SEARCH MICROBLOGS

| Chinese Word | English Translation |
|---|---|
| 堵, 拥堵 | Congestion |
| 车祸 | Traffic accident |
| 剐蹭 | Sideswipe |
| 事故 | Accident |
| 绕行 | Detour |
| 追尾, 相撞 | Car Crash |
| 塞车 | Jam |
| 路况 | Real time traffic |

### B. Learning Word Embedding Representations

We use the CBOW model to learn word embedding representations for extracting traffic relevant microblogs. Each word in the corpus is represented with a word vector and the dimension of this vector is set to 200. The size of the context window is set to 5, namely utilizing 5 words to the left and to the right of a target word. To optimize computational efficiency, we applied hierarchical softmax technique.

### C. Performance Indexes of Classification Models

To evaluate the performance of the proposed deep neural network models, we employ three statistical metrics, which are precision, recall and F-measure. These indexes are commonly used in evaluating the classifiers. Table II defines these three indexes. As all these indexes are class specific, for a certain class with label $l$, true positive $(TP)$ represents the number of instances with label $l$ correctly assigned with label $l$, false positive $(FP)$ represents the number of instances with any other labels except $l$ incorrectly assigned with the label $l$, and false negative $(FN)$ represents the number of instances with label $l$ incorrectly assigned with any other labels exception $l$. Precision measures the exactness of a classifier and recall represents its effectiveness. To get a tradeoff between precision and recall, F measure is defined as the weighted harmonic mean of precision and recall. When $\beta$ equals to one, F measure is also called $F_1$ score, which is commonly used in evaluating classification models.

TABLE II
PERFORMANCE INDEXES

| Index | Definition |
|---|---|
| Precision | $pre = \frac{TP}{TP+FP}$ |
| Recall | $rec = \frac{TP}{TP+FN}$ |
| F measure | $F_\beta = (1+\beta^2)\frac{pre \cdot rec}{\beta^2 \cdot pre + rec}$ |

### *D. Determination of the Structure and Hyperparameters of a CNN model*

To obtain the best performance model, we ran grid search on the depth of convolutional layers, the pooling strategies, the number and length of filters in convolutional layers, and the number of epoch. We have experimented the CNN models with one, two and three convolutional layer and the following pooling operations were chosen from maximum pooling and average pooling. The filter number was chosen from $\{15, 50, 100, 150\}$ and the filter length was set from 2 to 6 with stride 1. Batch size was set 100 and epoch number started from $1$ to $20$. In addition, we have also evaluated the performance of model by adding a vanilla hidden layer below the top output layer. By running grid search, the model with one convolutional layer and 100 filters whose length is $4$ achieved the best score. The best architecture employed a maximum pooling layer and a vanilla hidden layer with 50 output units. The best performing structure and hyperparameters are summarized in Table III.

TABLE III
STRUCTURE AND HYPERPARAMETERS OF THE BEST PERFORMING CNN MODEL

| | |
|---|---|
| **Convolutional Layers** | 1 |
| **Filter Number** | 100 |
| **Filter Length** | 4 |
| **Pooling Strategy** | maximum pooling |

### *E. Results*

For comparison, we tested SVM model that was reported with good performance for text classification. One SVM model, named as bow-SVM, takes bag of unigram and bigram as input. And vecseq-SVM takes the feature matrix obtained in the subsection dataset description as input that is same for all other modes. Both two SVM models were trained using LIBLINEAR library [37], [38]. We also test multi-layer perceptron method with one hidden layer to compare with the proposed CNN method.

The detection results of traffic relevant class with different classification approaches are given in Table IV. For each approach, the experiments were conducted under tenfold cross-validation. As the F measure balances the precision and recall, we take $F_1$ measure as an example. The measures of CNN model achieved the highest score for traffic relevant class, which is $0.8983$. Among the competitive methods, bow-SVM achieved the best $F_1$ score. It is worth to mention that for traffic relevant class, the precision of bow-SVM model is the highest compared to that of all other methods. However this model consumes large amount of memory as the dimension of its feature space in this paper is $276,679$ which is nearly ten time over that of other three models.

TABLE IV
PERFORMANCES COMPARISON OF SVM, MLP AND CNN FOR TRAFFIC RELEVANT CATEGORY

| | $pre$ | $rec$ | $F_1$ |
|---|---|---|---|
| **bow-SVM** | $0.9095$ | $0.8727$ | $0.8908$ |
| **vecseq-SVM** | $0.8368$ | $0.7992$ | $0.8176$ |
| **MLP** | $0.8752$ | $0.8719$ | $0.8737$ |
| **CNN** | $0.8961$ | $0.9006$ | $\mathbf{0.8983}$ |

## V. CONCLUSIONS AND FUTURE WORKS

In this paper, we discuss the emerging social transportation topic on extracting traffic information from social media. Our paper provides novel methods for extracting traffic relevant information from Sina Weibo. Different from many existing approaches that are based on a small volume of data and use bag of n-gram language model, our methods utilize the word semantics and employ deep neural networks to learn the abstract features of microblogs based on a large dataset. We first train the word vectors with Skip-Gram model. Using the word embedding representations, we transform the manually labeled microblogs into feature vectors, which are used to train CNN models. We have experimentally show that the improvement of $F_1$ measure of extracting traffic relevant microblogs.

For future work, it would be interesting to further classifying traffic related microblogs into more categories like traffic accidents, traffic status, etc. Furthermore, inferring and reasoning traffic status by fusing social media data and traditional physical traffic detector data is also very attractive.

## ACKNOWLEDGEMENT

The authors would like to thank Pro. Pu Wang, Pro. Tingshao Zhu and Mr. Dongdong Jiao for their help with data collection and prepossessing.

## REFERENCES

[1] F. Y. Wang, "Parallel control and management for intelligent transportation systems: Concepts, architectures, and applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 630–638, Sept 2010.

[2] N. Zhang, F. Y. Wang, F. Zhu, D. Zhao, and S. Tang, "Dynacas: Computational experiments and decision support for its," *IEEE Intelligent Systems*, vol. 23, no. 6, pp. 19–23, Nov 2008.

[3] J. Zhang, F. Y. Wang, K. Wang, W. H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, Dec 2011.

[4] J. Zhou, C. L. P. Chen, L. Chen, and W. Zhao, "A user-customizable urban traffic information collection method based on wireless sensor networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1119–1128, Sept 2013.

[5] F. WANG, X. Wang, Y. Yuan, T. WANG, and Y. LIN, "Social computing and computational societies: The foundation and consequence of smart societies," *Chinese Science Bulletin*, vol. 60, no. 5-6, pp. 460–469, 2015.

[6] X. Wang, L. Li, Y. Yuan, P. Ye, and F.-Y. Wang, "Acp-based social computing and parallel intelligence: Societies 5.0 and beyond," *CAAI Transactions on Intelligence Technology*, 2016.

[7] X. Wang, X. Zheng, Q. Zhang, T. Wang, and D. Shen, "Crowdsourcing in its: The state of the work and the networking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1596–1605, 2016.

[8] P. Ye and D. Wen, "Optimal traffic sensor location for origin–destination estimation using a compressed sensing framework," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1857–1866, 2017.

[9] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*. IEEE, 2016, pp. 2663–2668.

[10] T. Sakaki, M. Okazaki, and Y. Matsuo, "Tweet analysis for real-time event detection and earthquake reporting system development," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 4, pp. 919–931, 2013.

[11] S. E. Middleton, L. Middleton, and S. Modafferi, "Real-time crisis mapping of natural disasters using social media," *IEEE Intelligent Systems*, vol. 29, no. 2, pp. 9–17, 2014.

[12] E. Aramaki, S. Maskawa, and M. Morita, "Twitter catches the flu: detecting influenza epidemics using twitter," in *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2011, pp. 1568–1576.

[13] E. D'Andrea, P. Ducange, B. Lazzerini, and F. Marcelloni, "Real-time detection of traffic from twitter stream analysis," *IEEE transactions on intelligent transportation systems*, vol. 16, no. 4, pp. 2269–2283, 2015.

[14] Y. Gu, Z. S. Qian, and F. Chen, "From twitter to detector: Real-time traffic incident detection using social media data," *Transportation research part C: emerging technologies*, vol. 67, pp. 321–342, 2016.

[15] J. Cui, R. Fu, C. Dong, and Z. Zhang, "Extraction of traffic information from social media interactions: Methods and experiments," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 1549–1554.

[16] C. Gutierrez, P. Figuerias, P. Oliveira, R. Costa, and R. Jardim-Goncalves, "Twitter mining for traffic events detection," in *Science and Information Conference (SAI), 2015*. IEEE, 2015, pp. 371–378.

[17] P. Tejaswin, R. Kumar, and S. Gupta, "Tweeting traffic: Analyzing twitter for generating real-time city traffic insights and predictions," in *Proceedings of the 2nd IKDD Conference on Data Sciences*. ACM, 2015, p. 9.

[18] A. Kurkcu, M. Eng, E. F. Morgul, and K. Ozbay, "Extended implementation methodology for virtual sensors: Web-based real time transportation data collection and analysis for incident management," in *Transportation Research Board 94th Annual Meeting*, 2015.

[19] S. Zhang, J. Tang, H. Wang, and Y. Wang, "Enhancing traffic incident detection by using spatial point pattern analysis on social media," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2528, pp. 69–77, 2015.

[20] J. He, W. Shen, P. Divakaruni, L. Wynter, and R. Lawrence, "Improving traffic prediction with tweet semantics," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[21] M. Ni, Q. He, and J. Gao, "Using social media to predict traffic flow under special event conditions," in *Transportation Research Board 93rd Annual Meeting*, no. 14-3315, 2014.

[22] S. Grosenick, "Real-time traffic prediction improvement through semantic mining of social networks," Ph.D. dissertation, UNIVERSITY OF WASHINGTON, 2012.

[23] A. F. Abidin, M. Kolberg, and A. Hussain, "Improved traffic prediction accuracy in public transport using trusted information in social networks," in *Seventh York Doctoral Symposium on Computer Science & Electronics*, 2014, p. 19.

[24] A. F. Abidin, M. Kolberg, and A. Hussain, "Integrating twitter traffic information with kalman filter models for public transportation vehicle arrival time prediction," in *Big-Data Analytics and Cloud Computing*. Springer, 2015, pp. 67–82.

[25] M. Ni, Q. He, and J. Gao, "Forecasting the subway passenger flow under event occurrences with social media," *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[26] H.-P. Zhang, H.-K. Yu, D.-Y. Xiong, and Q. Liu, "Hhmm-based chinese lexical analyzer ictclas," in *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*. Association for Computational Linguistics, 2003, pp. 184–187.

[27] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, "Short text classification in twitter to improve information filtering," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp. 841–842.

[28] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[29] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of NAACL-HLT*, 2013, pp. 746–751.

[30] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[31] X. Rong, "word2vec parameter learning explained," *arXiv preprint arXiv:1411.2738*, 2014.

[32] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[33] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE transactions on neural networks*, vol. 8, no. 1, pp. 98–113, 1997.

[34] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[35] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[36] J. Bouvrie, "Notes on convolutional neural networks," 2006.

[37] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[38] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.