A Survey of Real-Time Social-Based Traffic Detection

Hashim Abu-gellban
Department of Computer Science
Texas Tech University, USA
hashim.gellban@ttu.edu

***Abstract*—Online traffic news web sites do not always announce traffic events in areas in real-time. There is a capability to employ text mining and machine learning techniques on the Twitter stream to perform event detection, to develop a real-time traffic detection system. In this present survey paper, we deliberate the current state-of-art techniques in detecting traffic events in real-time focusing on seven papers [1], [2], [3], [4], [5], [6], [7]. Lastly, applying effective text mining techniques and the SVM classifier in paper [2] gave the best results (i.e., 95.75% accuracy and 95.8% F1-score).**

***Index Terms*—Traffic detection, monitoring social networks, Twitter data stream.**

I. Introduction

The purpose of this survey paper was to detect and analyze traffic events by processing users' tweets in real-time. These tweets are belonged to a certain area and are written in different languages like English, Italian, and Thai. Additionally, the purpose was to support traffic and city administrations for managing scheduled or unexpected events in the city, to provide them with the ability to integrate this system to other traffic sensors (e.g., loop detectors, cameras, and infrared cameras) and to develop a monitoring system (e.g., detecting traffic difficulties). Other purposes were to provide a low-cost wide coverage of the road network, especially in those areas where traditional traffic sensors are missed and to develop a new Intelligent Transportation System (ITS).

Event detection from the analysis of the social network is a challenging problem than event detection from the traditional media because the social networks' texts are not well-formatted. For instance, the Status Update Messages "SUMs" (e.g., tweets information) are unstructured and irregular texts (i.e., it contains informal or abbreviated words, misspellings, or grammatical errors), have incomplete sources of information (i.e., it is very brief), and a large amount of useless information required filtering (e.g., over 40% of all tweets is pointless) [8].

This survey covers pioneer systems called ITSs that detect traffic events from social networks. These systems have some of the following features, which are the binary-classification (i.e., non-traffic, and traffic), and the multi-classification (e.g., non-traffic, congestion/crash traffic, and external events traffic). Also, they were built based on SOA architecture and traffic detection in tweets regarding a specific language. As tweets are unstructured and irregular with useless information, it is crucial to apply the dataset preprocessing using text mining techniques. We discussed the impacts of these techniques. Many classifications techniques were employed; whereas, the SVM classifier gave the best results. Also, the Cross-Validation method had been applied to evaluate the predictive models which were built by the classifiers. Generally, we will discuss and analyze the results to critique the techniques that previously had been used. These techniques might impact the results positively or negatively.

The importance of this research is coming from the audience's need of getting traffic information in real-time, while there is still a delay in traffic detection using the current monitoring systems and traffic sensors. It is important to note that the target audience of this research field is traffic news websites and radio news channels, police stations, and drivers.

Much research had been accomplished to extract useful information for event detection systems. These detection systems have two types: small-scale events and large-scale events. Concerning the small-scale event, it usually consists of a small number of Status Update Messages "SUMs" (i.e., shared user messages in social networks, where each message includes a text and meta-information, like timestamp and geographic coordinates). It also belongs to a precise geographic location in a short period, such as traffic, car crashes, fires, or local manifestations. Literature showed different ways of detection and classification of tweets for the small-scale system. For instance, Abel et al. [9] analyzed small-scale events like fires by extracting features using NER from Twitter streams and emergency network information. Whereas, Agarwal et al. [10] detected fires in a factory from Twitter stream analysis using standard NLP techniques and a Naive Bayes classifier.

On the other hand, a large-scale event is a huge number of SUMs and it has a wider temporal and geographic coverage like earthquakes and storms. Bansal et al. [11] presented a text mining analysis system to show the hottest topics in tweets at a specific time and in their places with an explanation regarding why these topics were interesting. Furthermore, Chew et al. [12] classified tweets of H1N1 containing related keywords and hashtags. Whereas, Sakaki et al. [13] detected earthquakes and typhoons in tweets by filtering specific keywords and employing SVM.

We presented the related work in Section II to give a brief about research in small-scale and large-scale events. In Section III, the methodologies, and technologies for de-

TABLE I: LIST OF LANGUAGES, TOOLS, CLASS LABELS, AND KEYWORDS

References	Language	Tools	Class Label	Keywords
1	English	Twitter's API and Crawling WEKA	1. traffic and posted by humans 2. traffic and posted by robots	1.Road Conditions aspects: ”holes”, ”flood”, ”pavement” 2.Delay Times aspects: ”slow”, ”stop”, ”jam” 3.Tolls or Fares aspects: ”cheap”, ”costly”
			3. Non-traffic	
2	Italian	Twitter's API and Crawling WEKA	1. Traffic.	“traffico” (traffic) or “coda” (queue) or “incidente” (crash)
			2. Non-traffic	
			1. Traffic due to external event	1. (“traffico” (traffic) or “coda” (queue)) and “partita”(match) 2. (“traffico” (traffic) or “coda” (queue)) and “processione” (procession) 3. (“traffico” (traffic) or “coda” (queue)) and “concerto” (concert) 4. (“traffico” (traffic) or “coda” (queue)) and “manifestazione” (demonstration)
			2. Traffic congestion or crash	1. “traffico” (traffic) and “incidente” (crash) 2. “traffico” (traffic) and “coda” (queue) 3. “incidente” (crash) and “coda” (queue)
			3. Non-traffic	
3	English	Twitter's API and Crawling WEKA	1. Traffic.	“vehicle”, “accident”, “road”, “collision”, “crash”, “wreck”, “injury”, “fatal accident”, “casualty”.
			2. Non-traffic	
4	English	Twitter's API	1. Traffic.	e.g. “car accidents”
			2. Non-traffic	
5	Thai	Twitter's API	1. Traffic.	e.g. “รถติด” (traffic congestion) and “อุบัติเหตุ” (accident)
			2. Non-traffic	
6	Arabic	Twitter's API	1. Fire	e.g., ‘# ’ان ا_جـده meaning (#Jeddah_now), ‘##) ان ا_ايا Riyadh_now).
			2. Weather	
			3. Social	
			4. Traffic Condition	
			5. Roadwork	
			6. Road Damage	
			7. Accident	
			8. Road Closures	
7	Portuguese	Twitter's API and MEKA	1. Incidents	N/A
			2. Weather Condition	
			3. Breakdown	
			4. Traffic Lights	
			5. Light Traffic	
			6. Heavy Traffic	

TABLE II: CLASSIFIERS AND THEIR BEST RESULTS

References	Class Label	Evaluation of Dataset	Classifiers	Best Results			
				Accuracy	Precision	Recall	F1-Score
1	3	Manual	SVM	N/A	54.10%	15.40%	24.00%
2	2	10-Fold Cross Validation	SVM, C4.5, KNN, NB, PART	95.75% (SVM)	95.80% (SVM)	96.25% (SVM)	95.80% (SVM)
	3		SVM, C4.5, KNN, NB, PART	88.89% (SVM)	89.60% (SVM)	88.90% (SVM)	88.97% (SVM)
3	2	10-Fold Cross Validation	SVM, NBB, JRip	90.24% (SVM)	90.40% (SVM)	90.20% (SVM)	90.20% (SVM)
4	2	N/A	Linear Regression Model	80.00% (for all incidents)	N/A	N/A	N/A
5	2	N/A	Syntactic Analysis	93.23%	62.77%	95.36%	75.70%
6	8	N/A	SVM, LR, and NB	90.00% (SVM)	88.00% (SVM)	89.00% (SVM)	90.00% (SVM)
7	6	2-Fold Cross Validation	RAkEL, BR, BCC, CT, and PCC	83.20% (RAkEL)	82.50% (RAkEL)	92.30% (RAkEL)	87.00% (RAkEL)

tecting real-time traffic events from Twitter data streams are described and categorized according to the tools, preprocessing of the datasets, classifiers (e.g., SVM), evaluations (e.g., k-fold Cross-Validation), and general discussion to analyze the results. Section IV provides a conclusion. Finally, future work is discussed in Section V.

II. Related Work

a) ***small-scale events:*** Wang et al. [14] used tweets and the semisupervised Latent Dirichlet Allocation (LDA) algorithm for their instant traffic alert system called “tweet-LDA”. They used keywords (e.g., M25, traffic, or jam) to crawl tweets related to traffic employing Twitter APIs in the UK. Lin et al. [15] proposed a neural network (NN) architecture to build a hierarchical TAPI classification model through analyzing the prediction of the most congested condition with a combination of the duration of congestion levels. They designed a shallow NN architecture with two layers and two activation functions for a dataset extracted from a Navigation System. Using deep learning [16] may improve the performance to detect traffic events, especially for high-dimensional data.

b) ***large-scale events:*** Nguyen et al. [17] developed a new approach called NiPred to predict people's needs for Hurricane Disaster relief using a One-Versus-Rest multi-class classification algorithm based on spatial-temporal tweets. Du et al. [18] compared Twitter and News to analyze people's concerns during the California wildfire event in 2018. They used the Latent Dirichlet Allocation (LDA) algorithm to identify the topics on Twitter during the wildfire.

In this paper, we studied seven papers on traffic detection based on tweets (i.e., SUMs). The traffic detection system is categorized as a small-scale events system. In this research, we compared the results of these papers with each other and discussed the key issues of the employed techniques.

III. Real-Time Traffic Detection in Twitter

We studied seven research papers related to traffic detection using Twitter [1], [2], [3], [4], [5], [6], [7]. TABLE I and TABLE II shows the tools and methodologies used in the seven recent research papers and their results. Some of the information is not mentioned in some papers; therefore, we use (N/A) in this case. In this Section, we discussed the tools which were been used in these papers and the methodologies of preparing the training datasets. Also, we deliberated classifications methods, the evaluations of the predictive classifier models, and the analysis of results. Finally, our survey provides some pros and cons of these techniques in order to explain the key reasons for their good and unpleasant results.

A. Tools

Twitter's API was used to crawl relevant traffic tweets [1], [2], [3], [4], [5], [6], [7]. This API was employed to directly access and fetch the public stream of tweets with specific filtering according to date, time, geolocation, language, and keywords as shown in TABLE I. Kokkinogenis et al. [1],

TABLE III: FEATURES

References	Features / Feature Approaches
[1]	Unigram Bag-of-Words
[2]	IDF
[3]	Word N-Grams, Char N-grams, TF-IDF, and (Syntactic, Spatial/Temporal, FeGeLOD) features
[4]	Content, User, and Usage features
[5]	Lexto, Spatial (Road and Start/ End points), Temporal, and Google Geocoding
[6]	TF-IDF
[7]	Vector Space Model, and Term Frequency Variance Index

Schulz et al. [3], and Li et al. [4] filtered tweets according to the English language; whereas, D'Andrea et al. [2] and Wanichayapong et al. [5] focused on Italian, and Thai languages, respectively. The authors of these papers filtered tweets according to several different keywords to crawl the tweets in a way that would allow an easy classification of them. For instance, Kokkinogenis et al. [1] employed the keywords (i.e., holes, flood, and pavement) to get "Road Conditions" tweets. Whereas, D'Andrea et al. [2] used "traffico" (traffic), "coda" (queue), or "incidente" (crash) to filter traffic class labels in Italian tweets. Moreover, D'Andrea et al. [2] used Twitter4J library. This library is to wrap the Twitter API, to be called through Java programs in ITS.

Kokkinogenis et al. [1], D'Andrea et al. [2], and Schulz et al. [3] used WEKA [19] (Waikato Environment for Knowledge Analysis) [9] to classify the extracted features from tweets. They employed the LIBSVM (i.e., integrated software for Support Vector Machine "SVM" classification) which is available through WEKA. LIBSVM is required to be downloaded first; then, the classpath must be modified to access "libsvm.jar" which is included in LIBSVM distribution. Whereas, Li et al. [4] and Wanichayapong et al. [5] used their programs to classify tweets using a linear regression model (LR) and syntactic analysis techniques, respectively. Chamby-Diaz et al. [7] used MEKA (i.e., a toolbox for classification) with default parameters.

B. Preprocessing

Event detection from social networks analysis requires preparing the data before classifying. This is because SUMs are unstructured and very brief. Also, they contain irregular texts and some useless information that is required to be filtered. Therefore, the authors [1], [2], [3], [4], [5], [6], [7] employed several text mining techniques to prepare the data in a more formal way to be efficiently classified to give good results. Some of these techniques were used in many of these papers, while others were just employed in one paper.

Kokkinogenis et al. [1] extracted location using Named Entity Recognition (NER), and Entity Linking (EL) methods. NER is employed to extract references of locations from the tweets' messages. Whereas, EL is used to distinguish these references from each other as two streets may have the same name in different cities. However, NER and EL [20], [21] are more challenging in texts (e.g., tweets), which are informal and short. As a result, the authors got an unpleasant result (i.e., 24% F1-Score) as shown in TABLE II. Furthermore, they filtered the stop-words by removing words that did not provide useful information such as articles, conjunctions, and noisy words. This filter helps to minimize the impacts of useless information in the analysis, which may provide incorrect classification.

D'Andrea et al. [2] used 5 various efficient text mining techniques to have a good classification by formalizing the text in a good way. First, the tokenization technique was employed in tweet texts to transform them into a stream (i.e., tokens) of syllables, words, and phrases. Second, the stop-word filtering method was used on the tokens to remove useless information. Third, the stemming technique was applied to transform each token to its root form by eliminating its suffix. Fourth, the stem filtering technique was employed to remove any stem which did not belong to the set of relevant traffic stems called "F relevant stems". The last method was called feature representation to assign a weight for every relevant stem or word in the feature vector. The F relevant stems and their weights were extracted and computed during the supervised learning stage according to their importance (i.e., frequent occurrences) in the dataset using the Inverse Document Frequency (IDF) index. These text mining techniques produced robust good results (e.g., 95.75% accuracy and 95.8% F1-score) for 2-class label classification and (e.g., 88.89% accuracy and 88.97% F1-score) for multi-class label classification. Whereas, multi-class label classification is more challenging work since it requires to distinguish two kinds of information (i.e., traffic due to external event, and traffic congestion or crash event) in tweets related to traffic events.

Schulz et al. [3] applied text mining techniques in tweets to prepare the data and extract features. The preprocessing phase includes seven processes. First, the pre-filtering method was applied to remove duplicate tweets that did not contain any additional useful information and removing "@-mentions" as were assumed to not be relevant to traffic detection. Second, eliminating the stop-words technique was applied. Third, correcting spelling errors using Google Spellchecking API was employed. Fourth, the slang replacement process was performed to transform abbreviations into formal words by using www.noslang.com. Fifth, Stanford POS tagger filtering was applied using "POS" software to extract proper nouns from the tweets. Sixth, the temporal mention replacement process was employed to distinguish chronological expressions, to eliminate the impact of overfitting the classification model. Lastly, the spatial mention replacement technique was employed to detect place and location.

Several feature extractions had been applied in the literature, where we will discuss only the features that gave the best results of this paper [3]. First, word-n-grams splitting was employed to divide a tweet text into a contiguous sequence of n words. For example, the word-3-grams of a tweet text like "Today the traffic is indeed unbelievable" is "Today the queue" and "is indeed unbelievable". Second, the syntactic features were extracted, e.g., the number of capitalized characters, "!", and "?" in a tweet text. Finally, the accumulated TF-

IDF score feature was extracted to measure the reflection of the importance of a tweet compared to all positive tweets in the training dataset. Therefore, using these proper text mining techniques for preprocessing and features' extractions helped giving good results (i.e., 90.24% F1-Score).

Li et al. [4] extracted temporal and spatial information from the tweets and stored them in the database. They used several methods to extract features. First, a traffic URL feature was extracted in such a way that if a tweet contained a URL to a news website then it might have been related to traffic conditions. The second feature is the similarity of tweets to measure the frequency of a tweet with other similar tweets within time and geolocation ranges. The third feature is hashtag frequency which is the number of tweets that have the same hashtag used within a tweet. The last one is traffic specific features to assign a "Yes" value if the tweet contains time, number, and geolocation otherwise that will be a "No". These techniques gave a fair result (e.g., 80% accuracy) where this result was accumulative with all other kinds of detections (e.g., traffic) since this paper was a general-purpose detection.

Wanichayapong et al. [5] used different methodologies of preprocessing for the tweets. First, they created a special dictionary to categorize every traffic word into a place, verb, ban word, or prepositions. Then, every word in a tweet was tokenized and categorized using this dictionary. After that, a heuristic filtering system was employed to extract real traffic tweets. To be considered as traffic-related tweets, the tweets must contain places and verb words according to this specific dictionary, not containing any ban word, and not having question words. Next, start and end attributes were extracted by using start and end preposition, with places from the tweet texts instead of geolocation inside the tweet's metadata. These techniques did not give robust good results (e.g., 75.7% F1-score). This was because extracting place from the tweet text might not be a proper way since the tweets were informal and might not refer accurately to the location of the events. Additionally, Alomari et al. [6] removed irrelevant characters from tweets (e.g., the punctuation symbols, "#" from hashtags, stop words), and three types of diacritics. They applied tokenization and normalization algorithms. They built a new stemming algorithm called "Iktishaf". Furthermore, Chamby-Diaz et al. [7] deleted the word "RT", user mentions "@", URLs, punctuation, stop words, numerical characters, and special characters (emoticons). They did not use any stemming algorithms and composed a feature set based on the weights of the keywords in a tweet. Finally, Table III shows the features and features approaches used in the seven papers. These features were extracted during the preprocessing datasets.

C. Classification

Kokkinogenis et al. [1], D'Andrea et al. [2], Schulz et al. [3], and Alomari et al. [6] applied the SVM classifier that gave better results than other classifiers used in these three papers. SVM is a popular classifier which is defined by a separating hyper-plane. The output of this algorithm is the optimal hyperplane that categorizes new examples. This algorithm can handle high-dimensional vector spaces, which makes feature selection less critical.

Kokkinogenis et al. [1] employed human users, and robot users to generate the training dataset with formal messages similar to tweet texts to easily classify it. Then, SVM was employed for classification. However, using a robot was not a good idea as real tweets were informal and short. This approach did not give good results (e.g., 24% F1-score).

SVM, C4.5, KNN, NB, and PART classification algorithms were employed by D'Andrea et al. [2]; whereas, SVM, NBB, JRip were employed by Schulz et al. [3]. Li et al. [4] applied LR Model. The syntactic analysis was used by Wanichayapong et al. [5] to classify traffic tweets and specify geolocation according to the start and end attributes by using Google Geocoding API. Moreover, Alomari et al. [6] applied SVM, the Logistic Regression (LR), and the Naïve Bayes (NB) algorithms. While, Chamby-Diaz et al. [7] employed 6 different classification algorithms (Random k-Labelsets "RAkEL", Binary Relevance, Classifier Chain, Bayesian Chain Classifier, Classifier Trellises, and Probabilistic Classifier Chain). They found that RAkEL provided the best performance.

The training time to build a classifier model using SVM employed in the first three papers [1], [2], [3] is slower than LR used in the fourth [4]. The time complexity of SVM (LibSVM) is $O(n^3)$, where n denotes the number of examples. The time complexity of LR is $O(nd^2 + d^3)$ and the space complexity is $O(nd + d^2)$ floats, where d is the number of features. The time complexity of the last paper [7] is $O(mg(|C|, n, d))$ [22], where m is the number of models, $|C|$ is the number of class labels, and g is the time complexity of the used classifier.

D. Evaluation

TABLE II shows the evaluation of the predictive models. Kokkinogenis et al. [1] used a manual process to evaluate its experiment. D'Andrea et al. [2] and Schulz et al. [3] employed 10-fold Cross-Validation to evaluate the classifiers. 10-fold cross-validation is commonly used to split the datasets. The datasets are randomly partitioned into 10 approximately equally sized subsamples. Subsequently, 10 iterations of training data and test data are performed such that within each iteration a fold of the data is used once for testing while the remaining 9 folds are used for training datasets. This approach is very important if the amount of data is small (e.g., small-scale events). This helps traffic events' experiments to reduce the sensitivity of the partitioning. Whereas, Li et al. [4] and Wanichayapong et al. [5] did not mention the way that they evaluated predictive models.

Four different measurements (i.e., accuracy, precision, recall, and F1-score) were used to evaluate the correctness and performance of the classifiers. These evaluations depend on four values, viz., True Positive "TP", False Positive "FP", True Negative "TN", and False Negative "FN".

Accuracy was used to measure the accuracy of predicting the class label. The accuracy measure may not be well suited for evaluating models derived from an imbalanced

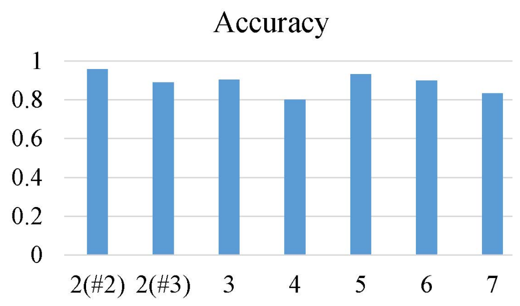

Fig. 1: The accuracy of the 7 papers.

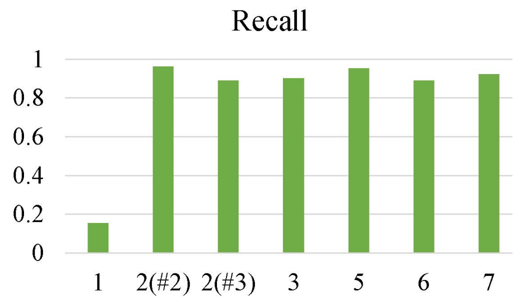

Fig. 3: The recall results of the 7 papers.

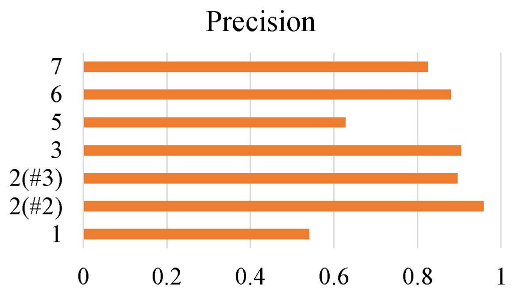

Fig. 2: The precision results of the 7 papers.

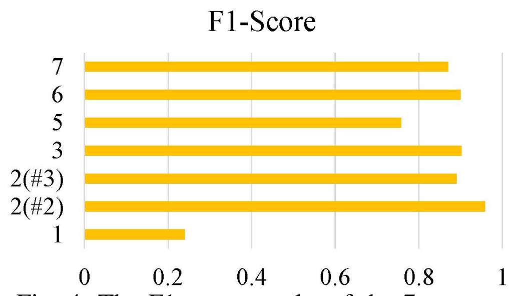

Fig. 4: The F1-score results of the 7 papers.

dataset since the positive or negative class labels are high or low. The accuracy formula is as follows: $Accuracy = \frac{TP+TN}{TP+FP+FN+TN}$

Precision (i.e., positive predictive value) was used to measure the ratio of relevant test instances, as the following formula: $Precision = \frac{TP}{TP+FP}$

Recall (i.e., sensitivity) is the ability of the test to correctly detect positive instances, as the following formula: $Recall = \frac{TP}{TP+FN}$

F1-score is a measurement that takes into consideration both precision and recall to make a good judgment about the results especially in an imbalanced dataset, as the following formula: $F1-score = \frac{2*Precision*Recall}{Precision+Recall}$

E. Analysis

Here, we discussed the pros and limitations of the proposed approaches. We analyzed the performance of the best methods in all papers [1], [2], [3], [4], [5], [6], [7] using the SVM classification method, the linear regression model, the syntactic analysis, and the RAkEL algorithm over two or more class labels. We found out that the SVM method in paper [2] with a 2-class label had delivered the best results of the four measurements as shown in Fig. 1, Fig. 2, Fig. 3, Fig. 4, and TABLE II. This might be related to that the authors followed the proper text mining techniques. This was done by using useful information on the tweets to formulate a proper weight for every root word according to its occurrences in the training dataset. Also, they used a balanced training dataset in supervised learning to build the model. Furthermore, they manually checked the class labels of every tweet in the training dataset and found that 4% of tweets had the keywords of the positive class label, but these tweets were actually negative instances. Therefore, they set the class label for these tweets as a negative class label. The IDF method used in [2] focuses on each rare term over the whole tweets (i.e., document collection) rather than the frequency of a term in the tweet, to penalize the common words. IDF provides the classifier with useful features, to enhance the performance.

The most unpleasant results were in paper one [1] as shown in Fig. 2, Fig. 3, Fig. 4, and TABLE II. This was because of the use of inefficient methods to extract features. For instance, NER and EL methods were known to suffer from informal and short texts as in tweets [20], [21]. Likewise, the authors did not apply state-of-the-art techniques such as stem filtering and stem weighting. Furthermore, transforming abbreviations into formal words and detecting the locations used in [3] are challenging problems that might produce misinformed information (features). In this case, the preprocessing might impact the performance of the classifier. Also, it might be better to drop abbreviations and tweets without geolocation. Additionally, using only the news websites' URLs in [4] to filter the related traffic tweets instead of using useful keywords for filtering like "traffic" is not enough since this approach discards crucial tweets which impact negatively on the performance of ITS. Moreover, Alomari et al. [6] applied "Iktishaf" (i.e., a new stemming algorithm); however, the accuracy of the NB model was dropped one percent after using their proposed stemming algorithm, while the performance of the other classification algorithms (SVM and LR) did not change after using Iktishaf. In general, the new Iktishaf algorithm did not improve the performance for the classification algorithms by using the Arabic dataset. This is because there is not enough research for Arabic text mining, as for other languages like English [23]. For instance, a simple stemming in English can be implemented by deleting of suffixes in Information

Retrieval. Whereas, the root-based stemming or the light stemming algorithm in Arabic needs to manipulate the four different types of affixes (i.e., prefixes, suffixes, postfixes, and antefixes). Finally, some key features might be not presented in the feature set in [7] since it was based on keywords rather than the key stems which have been used in [2]. Increasing the action space (i.e., more than two class labels) might have negatively affected the performance of the classifiers in the papers [1], [2], [6], [7].

IV. Conclusion

This survey paper focused on a deliberate summary view of the current state-of-art in traffic detection events. We used the latest related papers [1], [2], [3], [4], [5], [6], [7] in the traffic detection field. The goal of these papers was to build ITSs for real-time detection of traffic-related events from the Twitter stream. They might also notify users about the presence of traffic events. These papers used many text mining and machine learning techniques to classify tweets. Some authors employed state-of-the-art techniques like D'Andrea et al. [2], in order to detect traffic events often before online news web sites and local newspapers. Moreover, the results in the second paper [2] were the best among all the other papers (i.e., 95.75% accuracy and 95.8% F1-score) using adequate and effective preprocessing techniques (e.g., stem filtering and IDF) as well as the SVM classification algorithm. In general, the preprocessing phase using text mining techniques is crucial to enhance the classifiers.

V. Future Work

We may see a better detection system in real-time social networks in the future. For instance, Facebook is more popular than Twitter, which might improve the performance of the detection system and decrease the response time. Additionally, using new classification approaches like the Deep Learning (DL) with good neural network architecture might enhance the performance of the detection systems. This approach might help in several cases, e.g., if there is not enough research in text mining for some languages like Arabic since DL can extract features and when manipulating the unstructured and irregular texts. DL can also handle a large state space (large feature set) since it has been implemented successfully in many areas like image processing, time series, and text classification.

VI. Acknowledgment

I thank Professor Wen Xu for helping and supporting me in social computing techniques and methodologies. Finally, I thank my family, brothers, and sisters for all their support and encouragement.

References

[1] Z. Kokkinogenis, J. Filguieras, S. Carvalho, L. Sarmento, and R. J. Rossetti, "Mobility network evaluation in the user perspective: Real-time sensing of traffic information in twitter messages," in *Advances in Artificial Transportation Systems and Simulation*. Elsevier, 2015, pp. 219–234.

[2] 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.

[3] A. Schulz, P. Ristoski, and H. Paulheim, "I see a car crash: Real-time detection of small scale incidents in microblogs," in *Extended semantic web conference*. Springer, 2013, pp. 22–33.

[4] R. Li, K. H. Lei, R. Khadiwala, and K. C.-C. Chang, "Tedas: A twitter-based event detection and analysis system," in *2012 IEEE 28th International Conference on Data Engineering*. IEEE, 2012, pp. 1273–1276.

[5] N. Wanichayapong, W. Pruthipunyaskul, W. Pattara-Atikom, and P. Chaovalit, "Social-based traffic information extraction and classification," in *2011 11th International Conference on ITS Telecommunications*. IEEE, 2011, pp. 107–112.

[6] E. Alomari, I. Katib, and R. Mehmood, "Iktishaf: a big data road-traffic event detection tool using twitter and spark machine learning," *Mobile Networks and Applications*, pp. 1–16, 2020.

[7] J. C. Chamby-Diaz and A. Bazzan, "Identifying traffic event types from twitter by multi-label classification," in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*. IEEE, 2019, pp. 806–811.

[8] J. Weng and B.-S. Lee, "Event detection in twitter." *Icwsm*, vol. 11, no. 2011, pp. 401–408, 2011.

[9] F. Abel, C. Hauff, G.-J. Houben, R. Stronkman, and K. Tao, "Twitcident: fighting fire with information from social web streams," in *Proceedings of the 21st International Conference on World Wide Web*, 2012, pp. 305–308.

[10] P. Agarwal, R. Vaithiyanathan, S. Sharma, and G. Shroff, "Catching the long-tail: Extracting local news events from twitter," in *Sixth international AAAI conference on weblogs and social media*, 2012.

[11] N. Bansal and N. Koudas, "Blogscope: spatio-temporal analysis of the blogosphere," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 1269–1270.

[12] C. Chew, "Pandemics in the age of twitter: A content analysis of the 2009 h1n1 outbreak," Ph.D. dissertation, Citeseer, 2010.

[13] 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, 2012.

[14] D. Wang, A. Al-Rubaie, S. S. Clarke, and J. Davies, "Real-time traffic event detection from social media," *ACM Transactions on Internet Technology (TOIT)*, vol. 18, no. 1, pp. 1–23, 2017.

[15] Y. Lin and R. Li, "Real-time traffic accidents post-impact prediction: Based on crowdsourcing data," *Accident Analysis & Prevention*, vol. 145, p. 105696, 2020.

[16] H. Abu-gellban, L. Nguyen, M. Moghadasi, Z. Pan, and F. Jin, "Livedi: An anti-theft model based on driving behavior," in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, 2020, pp. 67–72.

[17] L. H. Nguyen, S. Jiang, H. Abu-gellban, H. Du, and F. Jin, "Nipred: Need predictor for hurricane disaster relief," in *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*, 2019, pp. 190–193.

[18] H. Du, L. Nguyen, Z. Yang, H. Abu-gellban, X. Zhou, W. Xing, G. Cao, and F. Jin, "Twitter vs news: Concern analysis of the 2018 california wildfire event," in *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2. IEEE, 2019, pp. 207–212.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[20] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, 2009, pp. 147–155.

[21] S. Cucerzan and D. Yarowsky, "Language independent named entity recognition combining morphological and contextual evidence," in *1999 joint SIGDAT conference on empirical methods in natural language processing and very large corpora*, 1999.

[22] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *European conference on machine learning*. Springer, 2007, pp. 406–417.

[23] H. A. Almuzaini and A. M. Azmi, "Impact of stemming and word embedding on deep learning-based arabic text categorization," *IEEE Access*, vol. 8, pp. 127 913–127 928, 2020.