Computational Rumor Detection Without Non-Rumor: A One-Class Classification Approach

Amir Ebrahimi Fard, Majid Mohammadi, Yang Chen, *Senior Member, IEEE*, and Bartel Van de Walle

***Abstract*— Rumor spreading in online social networks can inflict serious damages on individual, organizational, and societal levels. This problem has been addressed via computational approach in recent years. The dominant computational technique for the identification of rumors is the binary classification that uses rumor and non-rumor for the training. In this method, the way of annotating training data points determines how each class is defined for the classifier. Unlike rumor samples that often are annotated similarly, non-rumors get their labels arbitrarily based on annotators' volition. Such an approach leads to unreliable classifiers that cannot distinguish rumor from non-rumor consistently. In this paper, we tackle this problem via a novel classification approach called one-class classification (OCC). In this approach, the classifier is trained with only rumors, which means that we do not need the non-rumor data points at all. For this study, we use two primary Twitter data sets in this field and extract 86 features from each tweet. We then apply seven one-class classifiers from three different paradigms and compare their performance. Our results show that this approach can recognize rumors with a high level of F1-score. This approach may influence the predominant mentality of scholars about computational rumor detection and puts forward a new research path toward dealing with this problem.**

***Index Terms*— Non-rumor, one-class classification (OCC), rumor, rumor detection, tweet.**

I. Introduction

THE phenomenon of rumor propagation is one of the most severe challenges that societies are dealing with for many years. In fact, it is an ancient problem that can inflict damage by harming the reputation of individuals or organizations [1], shaking financial markets, and disrupting aid operations [2].

Traditionally, these kinds of information were propagated by means of word of mouth, newspapers, and radio. However, in recent years, the emergence and rapid growth of online social networks turned this problem into a major socio-political challenge due to the easy, fast, and wide propagation of information in online social networks. Because of volume and velocity of rumors in social networks, researchers use large-scale data and computational methods to control this phenomenon [1], [3]–[5]. One of the important steps in a rumor control system is rumor detection that has been a topic of interest for the community of computational social science in recent years. The main goal of rumor detection is to identify rumors in online social networks automatically, accurately, and in a timely manner.

Binary classification is the dominant computational approach for rumor detection [6]. In this approach, a model is built by training the classifier with a data set comprising samples of rumors and non-rumors. We then evaluate the discrimination power of the model by subjecting the trained classifier to a mixed set of rumors and non-rumors. The model with higher performance in separating rumors from non-rumors is considered as a better classifier. Although this is a well-established approach in the literature and several scholars used it for the computational rumor detection, it suffers from a serious flaw, namely, the non-rumor pitfall.

Non-rumor is a fuzzy concept that is widely used by scholars for computational rumor detection. Unlike the rumor that has a long-standing definition due to its solid background in social psychology, non-rumor has an ambiguous meaning, which is not supported by either epistemological or psychological studies. Ambiguity in this concept leads to different ways of data collection/annotation. In other words, the lack of a clear definition allows scholars to come up with their own readings of non-rumor [1], [7], [8]. For instance, Kwon *et al.* [1] considered any kind of factual information and news as non-rumor, while Zubiaga *et al.* [9] annotated tweets that are not rumor as non-rumor. This creates a confusing situation, because from one hand, there are some tweets that are not rumor, and on the other hand, they may or may not take the non-rumor label. Therefore, the model may receive data points that do not belong to its predefined classes. Those data points will randomly be classified in a binary classifier as rumor or non-rumor.

The implication of such discrepancy between various non-rumor definitions is that the binary classification is not the right approach for computational rumor detection. In other words, every scholar can define non-rumor according to which they collect data from social networks. This raises two important issues regarding rumor detection systems in real situations. First, the results of rumor classifiers become inconsistent, and

Manuscript received December 22, 2018; revised June 3, 2019; accepted July 14, 2019. This work was supported in part by the Engineering Social Technologies for a Responsible Digital Future Project at TU Delft and in part by the National Natural Science Foundation of China under Grant 61602122 and Grant 71731004. *(Corresponding author: Amir Ebrahimi Fard.)*

A. E. Fard, M. Mohammadi, and B. Van de Walle are with the Faculty of Technology, Policy and Management, Delft University of Technology, 2628 BX Delft, The Netherlands (e-mail: a.ebrahimifard@tudelft.nl; m.mohammadi@tudelft.nl; b.a.vandewalle@tudelft.nl).

Y. Chen is with the School of Computer Science, Fudan University, Shanghai 200433, China (e-mail: chenyang@fudan.edu.cn).

Digital Object Identifier 10.1109/TCSS.2019.2931186

second, the results of different classifiers cannot be compared. Therefore, the question that arises here is how can rumors be identified in social networks regardless of the non-rumor definition?

We address this question from the one-class classification (OCC) perspective [10]–[12]. OCC is a supervised algorithm with the ability to identify class X between multiple classes of data when the classifier is trained by class X only. For the computational rumor detection, this means that the classifier is trained with rumors only, and the trained classifier can be used to detect rumors from other kinds of tweets. In contrast to non-rumor, which is not well-defined and has a controversial conceptualization, there is a consensus in the literature for rumor, and scholars often follow definitions with similar elements.

In this paper, we benefit from two available data sets, one from [4] with *6425* tweets and the other from [1] with *140910* tweets. We extract 86 features from each tweet. After feature extraction, we build models with seven algorithms that implement an OCC approach. Each algorithm has two hyper-parameters that impact its performance. We use grid search over the hyper-parameters to discover the best performance of the models. Fig. 1 shows the research workflow in three stages: 1) data set preparation; 2) feature extraction; and 3) OCC.

The main contributions of this paper are summarized as follows:

1) discussing the conceptualization problem of non-rumor as a widely used term in the computational rumor detection;
2) addressing the weaknesses of the binary classification as the dominant approach in the computational rumor detection problem;
3) justifying the rumor detection as an OCC problem and designing a set of experiments over two data sets with seven prominent one-class classifiers.

The remainder of this paper is organized as follows. In Section II, we discuss the background of rumor studies and the computational rumor detection. Section III elaborates on the concept of non-rumor and the fallacies of the binary classification in the computational rumor detection. In Section IV, we discuss the general idea behind an OCC approach and explain seven renowned algorithms with this approach. Section V contains the features extracted for each tweet, while Section VI presents the results of the experiments regarding the rumor detection using OCC. Finally, we conclude the research by summarizing the main finding and giving some suggestions for future work in Section VII.

II. Background and Related Works

In this section, the background information and related works regarding the primary elements of this paper are provided. To give some context to the reader, we first introduce the field of rumor studies by briefly explaining the important theories and findings of the field. Then, we focus on computational rumor detection by first explaining the dominant approach of the rumor detection, and then reviewing some of the significant studies in the computational rumor detection, to give the state-of-the-art overview of this research domain.

A. Rumor Theories

Rumors are unverified propositions or allegations about an object, event, or issue, which are not accompanied by corroborative evidence [13], [14]. Rumors take different forms, such as exaggerations, fabrications, explanations [15], wishes, and fears [16]. Rumors have a life cycle and change over time. Allport and Postman [17] in their seminal work "psychology of rumor" concluded that "as a rumor travels, it grows shorter, more concise, more easily grasped and told." They refer to these three steps in rumor life cycle as leveling, sharpening, and assimilation. In the same vein, Buckner [18] considered rumor as a collective behavior which is becoming more or less accurate while being passed on as it is subjected to the individuals' interpretations.

Rumors tend to thrive and transmit in situations that are ambiguous and/or pose a threat or potential threat situations in which meanings are uncertain; questions are unsettled, information is missing, and/or lines of communications are absent [19]. Allport and Postman [17] gave a more formal interpretation of rumor transmission dynamic and called it basic law of rumor, that is

$$R \sim i \times a \tag{1}$$

where R, i, and a denote rumor intensity, importance, and ambiguity, respectively. This formula means that the intensity of a rumor varies with the importance of the rumor subject to the individuals times ambiguity of the evidence pertaining to the rumor topic. Hence, if a topic is not important for a certain community, there will not be any rumor about that topic in that community. Allport and Postman [17] provided an example on this topic that it is not likely for an American to spread rumors concerning the market price of camels in Afghanistan, because the subject has most probably no importance for him. Later, Rosnow [54] studied the influence of other factors, such as anxiety and credulity of the individuals on rumor transmission.

Rumors emerge to fulfill a particular function. Sometimes, rumors work as a diversion and serve the function of titillation and breaking the monotony. News is the other role of rumor, as certain people do not have access to any bona fide information concerning a situation. Need for meaning and structure or what the Gestalt theorists in psychology called "closure" is another situation that rumors emerge. Information verification and discovering the truth behind unverified pieces of information are another reason for spreading the rumor. Rumors can also play a political role and used by politicians to manipulate societies as Knopf [21] wrote that rumors "increase polarization while at the same time strengthening solidarity within each group." Rumors can also work as a defense mechanism for the ones who are unsuccessful or failed (see [22] for detailed explanations and examples).

Regardless of their roles, rumors cause serious issues [23]. They can damage the reputation of individuals or organizations [1], provoke riot and unrest [23], catapult firms into financial disaster [13], shake financial markets, and

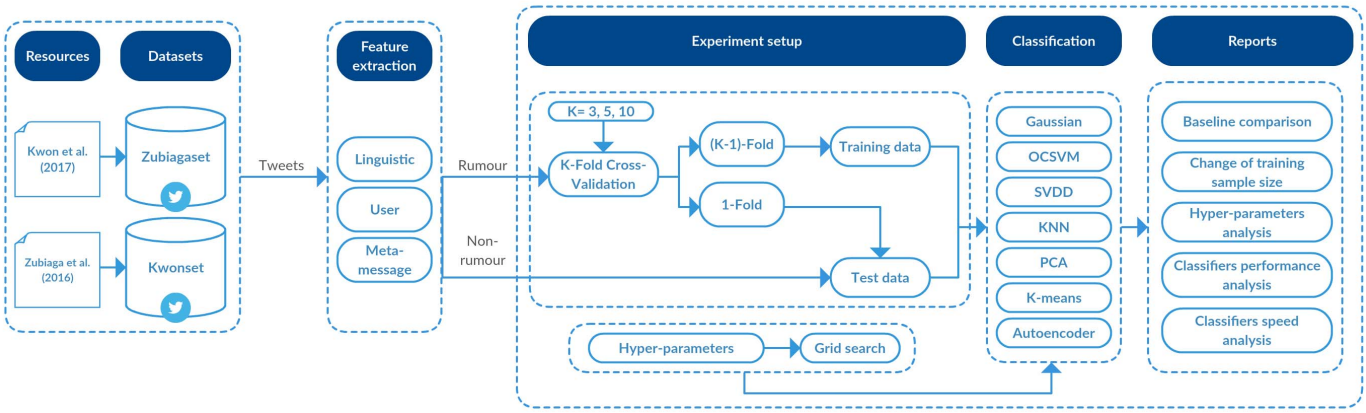

Fig. 1. Research flow of rumor detection with OCC approach.

disrupt aid operations [2]. Hence, it is necessary to manage this potentially harmful phenomenon effectively. DiFonzo and Bordia [23] provided a comprehensive list of rumor-quelling strategies drawn from the psychological and business literature. Modeling rumor propagation is the other approach toward rumor management, which is mostly developed by physicists and mathematicians on the basis of network models [24]–[27] and population dynamics [28]–[30]. This approach aims to provide an abstract realization of rumor propagation and examine the performance of different control strategies, such as source detection, network cut, and immunization. The other class of control strategies is a computational approach that attempts to tackle rumors using computational and statistical modeling. This approach has drawn much attention recently due to the rapid growth of online social networks that produced a tremendous amount of data. In Section II-B, we further elaborate on this approach.

B. Computational Rumor Detection

Computational solutions are among the most recent approaches that academia has adopted to tackle the problem of rumor spreading in social networks. This family of solutions is usually part of a framework called rumor resolution system that constitutes of four modules: 1) rumor detection, which specifies whether a piece of information is relevant to a rumor or not; 2) rumor tracking, which collects the posts discussing the rumor; 3) stance classification, which determines how each post orients to the rumor's veracity; and 4) veracity classification, which determines the truth behind the rumor [6]. The four modules are shown in Fig. 2.

In this paper, we focus on the detection module in Twitter, which aims to annotate tweets automatically as rumor or non-rumor to identify new rumor-relevant tweets that appear in Twitter. This problem has drawn computer scientists' attention in recent years due to its tremendous importance to the social context. In this problem, the main focus is on providing an accurate representation of rumor for the computer in such a way that it can recognize rumors with minimum error. Such a representation is a function f of three elements, i.e., data, features, and learning algorithm

$$\text{Quality of Rumor Classifier} \propto f(\text{data, features, learning algorithm}).$$

If rumors are modeled via sufficient data, illustrative features, and a powerful algorithm, the rumor detection system is expected to identify rumors accurately, but any flaw in those elements can impact the quality of the rumor detection. In the study of digital rumors, obtaining sufficient data is an arduous task due to the difficulties and barriers of knowing the latest rumors and restrictions of social networks over the data collection. Therefore, collecting and annotating data in different topics and from various social networks are considered as an important contribution to this field. For instance, Zubiaga *et al.* [8] created a data set of tweets, containing 6425 rumors and non-rumors, posted during five breaking news events. Sicilia *et al.* [7] also made a contribution by building a data set of Twitter rumors and non-rumors containing 709 samples regarding the Zika virus. In other work, Yang *et al.* [5] provided the first data set of Chinese microblogs from the biggest microblogging service in China, namely, Sina Weibo. In the same vein, Turenne [31] created the first French data set in this field by collecting 1612 rumor related texts. Additionally, in one of the biggest data sets in this field, Kwon *et al.* [32] provided a data set comprising more than 140 000 rumors and non-rumors in a wide range of topics from politics to entertainment.

The second important element of rumor detection is feature extraction. Rumors in their raw formats are incomprehensible for the computer. Hence, it is essential to find a way to make this concept machine-readable. The feature extraction aims to respond to the same need by selecting d quantifiable properties and representing each rumor with those properties to the computer. More formally, in this step, we map every data point (rumor) to a d-dimensional space, where each dimension represents one property of rumors. The better the features can reflect different dimensions of rumors to the computer, the more realistic the understanding of the computer from this phenomenon will be. Many of the computational rumor scholars studied new features and their impact on the rumor

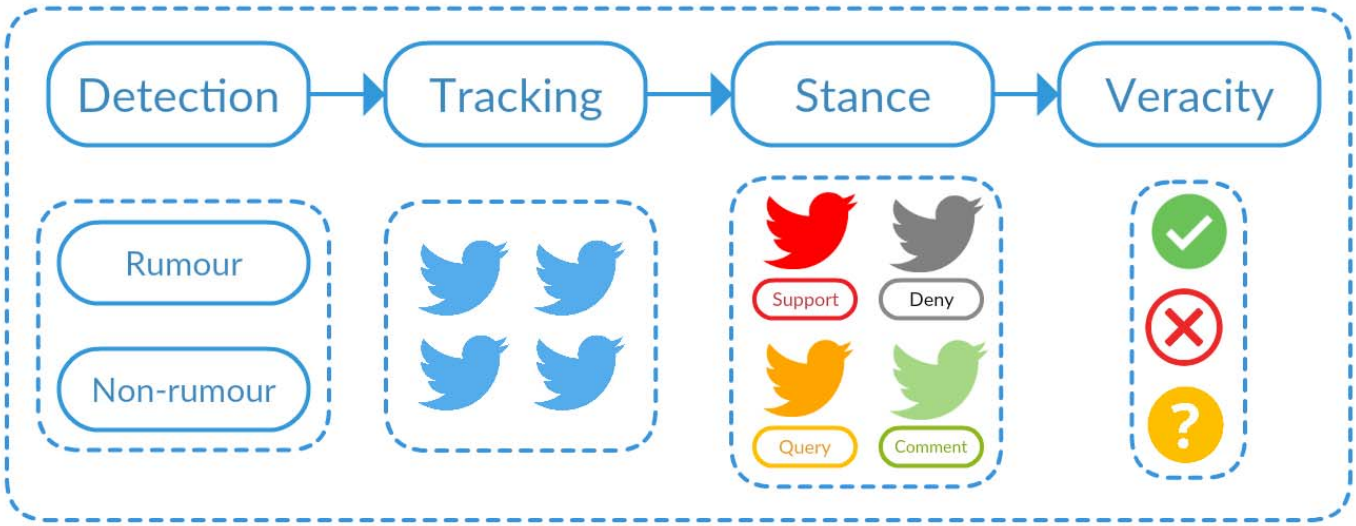

Fig. 2. Rumor resolution system has four modules: 1) rumor detection for identifying rumor related information; 2) rumor tracking for collecting the posts discussing the rumor; 3) stance classification for determining posts' orientations toward rumors' veracity; and 4) veracity classification for verifying truth behind the rumor [6].

detection. For instance, Castillo *et al.* [33] introduced the features related to the account holders and their social networks. In the other work, Kwon *et al.* [32] introduced temporal features and studied the importance of temporal patterns in rumor detection. In another work, Kwon *et al.* [1] discovered the effectiveness of features in detection of rumor that varies in the course of time. They concluded that some features (user and linguistic) are more effective for rumor detection in early stages of diffusion, while some others (structural and temporal) are more effective for rumor detection in longer time windows.

The third important element in computational rumor detection is the learning algorithm. In this step, a classifier defines its decision boundary to understand what rumor and non-rumor are. As the decision boundary gets more accurate, the classifier performance improves. Computational rumor researchers apply new algorithms, statistical analysis, or methods to this domain in order to detect rumors more accurately. To be considered as a contribution, these algorithms should not be necessarily newly designed. They might be popular algorithms in other contexts; however, they have not been used in rumor detection before. For instance, Ma *et al.* [34] applied recurrent neural networks (RNNs) to the rumor context for the first time. They evaluated their model with three widely used recurrent units, tanh, long short-term memory (LSTM), and gated recurrent unit (GRU), which could perform significantly better than state of the art. In the other work, Zubiaga *et al.* [4] modeled the rumor tweets as sequences by extracting their features in the course of time and applied the conditional random field (CRF). This allowed them to obtain the context from a set of tweets and identify rumors with higher performance.

III. Problem Statement

In Section II, we explained the binary classification as the dominant approach for computational rumor detection in the literature. However, the binary classification suffers from a major drawback for detecting rumors. In this section, we discuss the drawback and reason why binary classifiers cannot be appropriate for identifying rumors. To this end, we first explain the concept of non-rumor and provide some evidence of multiple ways of defining non-rumor in the literature. Then, in the second step, we argue how this controversial concept makes the binary classification unreliable and inconsistent for detecting rumors in real-world situations.

A. Concept of Non-Rumor in Computational Rumor Detection

As explained in Section II-B, we train the classifier with features obtained from rumor and non-rumor tweets. However, the question that remains here is "how do the data points take rumor/non-rumor labels?" Data points that are annotated define each class for the model.

For rumor, researchers most often refer to definitions with similar elements, which is due to years of research by rumor scholars that ultimately lead to relative convergence on some aspects of this field, such as rumor conceptualization. This allows researchers to annotate rumorous tweets consistently, but what about non-rumor? What is exactly the non-rumor? Are rumor and non-rumor complementary concepts, in a way that by specifying one, the other will be automatically specified? Or, they are not complementary, and there are other sorts of tweets that fall into neither rumor nor non-rumor category? These are non-trivial questions that, to the best of our knowledge, have not been addressed in the literature yet.

Non-rumor is an ambiguous term that is coined mostly by computer scientists who used the binary classification for detecting rumors. From a historical point of view, this term has been mentioned two times in the non-computational part of rumor literature [35], [36], but it has been defined in neither of them. Similarly, non-rumor has been used frequently in the literature of computational rumor detection; however,

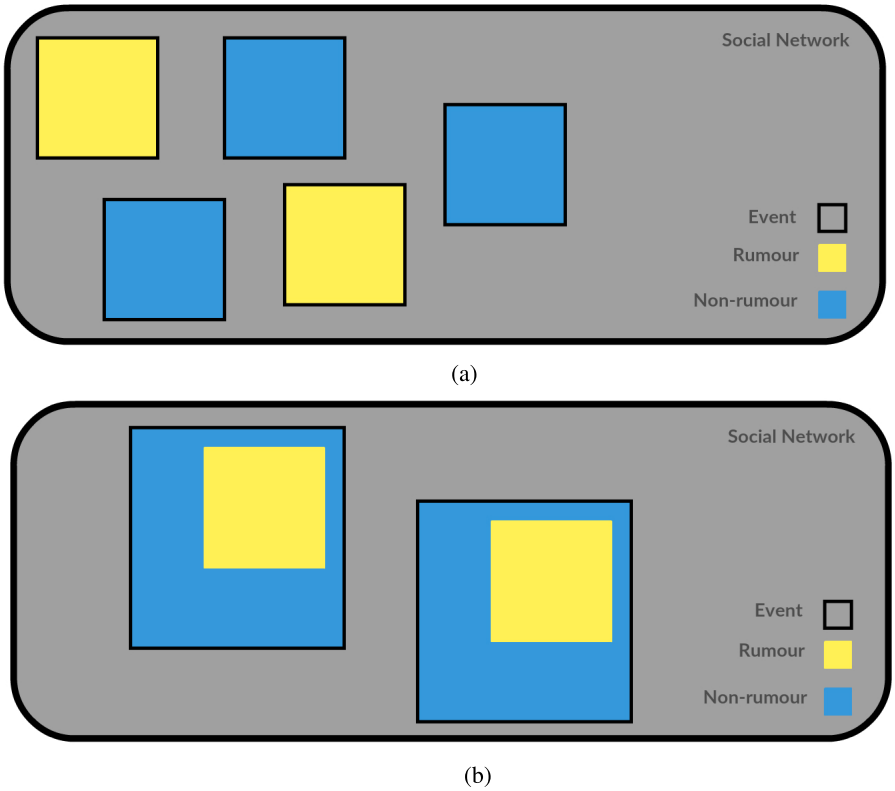

Fig. 3. Schematic description of two primary perspectives toward non-rumor. In both diagrams, squares with border show different events. Yellow and blue squares denote rumor and non-rumor area, respectively. Size does not mean anything and cannot be a basis for comparison. (a) Non-rumor as fact [1]. Five events are depicted in this diagram; two rumors worthy and three non-rumors worthy. (b) Non-rumor as any piece of information that cannot take rumor label [8]. Two events are depicted in this diagram. In each event, the yellow and blue areas show the rumor and non-rumor worthy part in each event.

it has not always been portrayed in the same way. In other words, there is no consensus between the researchers about the conceptualization of non-rumor. This leads to different ways of annotating the non-rumor tweets.

We investigated two primary and widely used data sets [1], [8] in the literature of the computational rumor detection which does not define non-rumor in the same way. Kwon *et al.* [1] took non-rumor as news items that are extracted from credible news sources, while Zubiaga *et al.* [9] treated non-rumors as any related information that cannot take the rumor label. Fig. 3 shows the conceptualization of rumor and non-rumor in these two data sets in a schematic way. In Fig. 3(a), which shows the idea of Kwon *et al.* [1] about rumor, the big gray rectangular area demonstrates the social network space. Each square in this area illustrates a distinct event. The ones with yellow color correspond to rumor worthy tweets, while the blue squares show tweets related to the reliable news. On the other hand, Fig. 3(b) shows the approach of Zubiaga *et al.* [8]. Like the previous one, the big gray area is the social network space, and each square with black border shows the relevant tweets to a particular event. However, unlike the approach of Kwon *et al.* [1] in which an event corresponds to either rumor or non-rumor, here an event is a mixture of rumor and non-rumors tweets. The yellow part of the squares shows the rumor relevant tweets, and the remaining blue area demonstrates the non-rumor tweets.

B. Fallacy of Binary Classification for Computational Rumor Detection

In this section, we argue how various interpretations of non-rumor can impact the quality of the rumor detection systems. To answer this question, we need to inspect the multi-class classification and specifically binary classification more closely to understand how they classify their input into predefined classes. To this end, we use the case of compote making factory to show how unexpected input can violate the consistency of the classification system.

We assume that a compote making factory produces five types of fruit compotes based on five different fruits (there is a one-to-one association between fruits and compotes). Because each type of compote is prepared in different parts of the factory, first of all, fruits must be categorized based on their type (at the beginning, the fruits are mixed). In fact, this machine is a multi-class classifier that is trained with samples from five types of fruits. We assume that any fruit that enters this machine belongs to one of the five pre-determined fruit types, but what would happen if this condition was violated? What would happen if once ten types, once five types, and once eight types of fruits entered to the machine? What would happen if new types of fruits entered to the machine with which it was not trained?

When a model is trained for k-classes and a new data point without belonging to any of these k-classes is given to

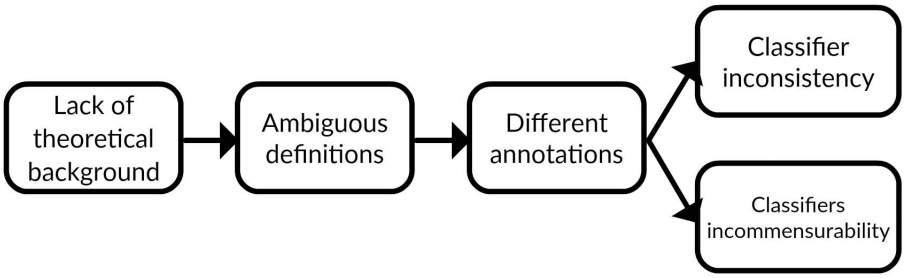

Fig. 4. Chain of the reasoning behind the problematic consequences of non-rumor in the binary classification. It starts with a lack of sufficient theoretical background for the concept of non-rumor. It leads to the emergence of ambiguous and contradictory definitions of non-rumor. Lack of clear definitions causes data annotation to be done arbitrarily, which makes the rumor classifier unreliable (it is not clear, what it separates) and incomparable (it is not possible to compare the results of different classifiers).

the classifier, the data point will be definitely classified into one of the k-classes. This is a case of false positive since that data point gets a wrong label. This is exactly similar to the case of the computational rumor detection using binary classification techniques. If we build a binary classification model for the rumor detection (according to any of the existing definitions for non-rumor), there might be data points coming to our system without belonging to the rumor or non-rumor classes (based on the definition we used). Hence, if we build a rumor detection system to identify rumors in the real world,[1] (depending on the definition of non-rumor we use to train the classifier), new data points that do not belong to rumor or non-rumor classes increase the number of false positives.

The lack of consensus on the meaning of non-rumor causes every researcher to come up with his/her definition. This diversity of definitions creates two major problems; first of all, the rumor detection becomes inconsistent and unreliable as we cannot be sure about their functions and what they separate. Second, we cannot compare the outcome of different models as they measure different things. Fig. 4 shows the chain of reasoning behind the problematic consequences of non-rumor in the binary classification.

Due to these difficulties, questions, and ambiguities in the classical rumor detection, we are thinking of the rumor detection with rumor data only. Such an approach can address both problems of the binary classification. It makes the classifier reliable and effective by identifying the relevant information to the rumor as rumor. From the data collection perspective, we can annotate rumor relevant data easily, something that is almost impossible for non-rumor. Therefore, in a data set, the annotation of rumors is a feasible task, while it is quite challenging for non-rumors. But how can we detect rumors without having and representing the other group of data? The answer is with an approach called OCC [12].

OCC is an approach that can address such a situation as it works based on the recognition of one class only and classifying it as a target class while annotating all the other data points as an outlier class. Table I shows the differences between the multi-class and on-class approaches. The second and fourth columns demonstrate different aspects of multi-class classification and OCC, respectively. The third column shows two situations that we use OCC: when the data set is imbalanced and when we are not sure about the number of classes. In Section IV, we discuss the OCC in more detail.

[1]It means in a non-experimental setting.

IV. One-Class Classification

Binary classification is the dominant strategy in the realm of pattern recognition and machine learning. Binary classifiers try to learn a function that is able to discriminate the samples of two given classes. To compute such a function based on the given samples, a plethora of methods exist, each of which has an underlying idea and is predicated on some presumptions. The classification problem compounds for the cases where we have the samples of one of the classes only, i.e., OCC problem. Several techniques have been proposed to solve an OCC problem. Tax [12] classifies one-class methods into three categories: the density estimation, the boundary methods, and the reconstruction methods. Fig. 5 shows an overview of the categories and their corresponding methods. In the following, several one-class classifiers for each of these categories are discussed.

A. Density Estimation

In this approach, the underlying idea is that target samples follow a distribution, which needs to be estimated in the training phase. The most prevalent distribution, which is used for the density estimation, is the multivariate Gaussian distribution. In this model, it is assumed that each of the training data $x \in R^d$ is a sample of a multivariate Gaussian distribution, that is

$$p_N(x; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\} \quad (2)$$

where $\mu \in R^d$ is the mean and $\Sigma \in R^{d\times d}$ is the covariance matrix. Thus, the training of this method entails the estimation of the mean and the covariance matrix. The maximum likelihood estimation (MLE) can swiftly estimate these parameters; thus, the OCC based on the density estimation is usually faster compared to other methods. Having estimated the distribution of the target samples, the probability that a test sample z belongs to the target class can be simply computed by the likelihood of z with respect to the estimated distribution. If the

TABLE I
COMPARISON BETWEEN THE MULTI-CLASS CLASSIFICATION AND THE OCC

	Multi-class Classification	**Transition**	**One-class Classification**
Number of classes	n≥2	Lack of either of the following conditions suffices for transition from multi-class to one-class classification: • well-represented class • well-identified class	2
Training dataset	A dataset comprising balanced samples of n classes		A dataset comprising only one class that we know
Test dataset	A dataset comprising balanced samples of n classes		A dataset comprising both classes
Training	Train the model with n classes		Train the model with one class
Test	Testing the model with n classes		Testing the model with two classes
Performance	Reporting the model performance via confusion matrix		Reporting the model performance via confusion matrix

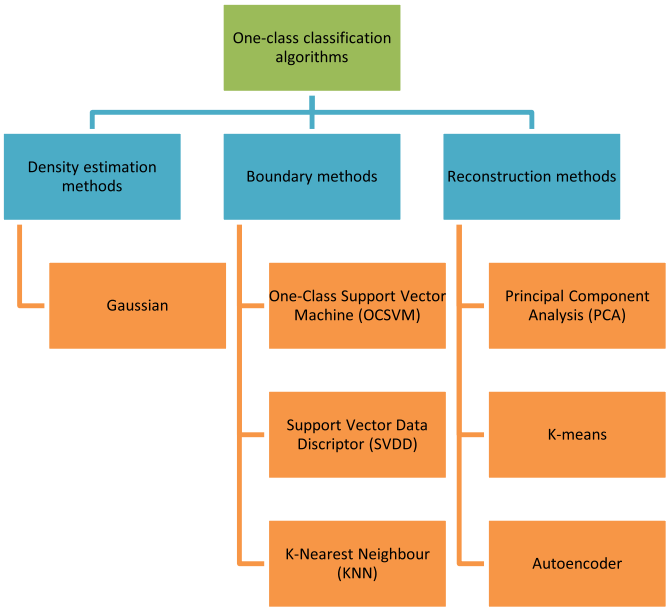

Fig. 5. Categorization of OCC algorithms.

computed likelihood is less than a threshold, then the sample z is said to be an outlier; otherwise, it belongs to the target class.

B. Boundary Methods

Due to limited data availability in some cases, the density estimation does not provide a comprehensive overview of the data and the result of the consequent one-class classifier is thus not acceptable. To tackle this problem, boundary methods are proposed, which tries to optimize a closed boundary around the target class. In this paper, we consider three algorithms with this perspective. The two widely used one-class classifiers are the one-class support vector machine (OCSVM) [37] and support vector data description (SVDD) [38]. Given a set of training samples $\{x_i\}_{i=1}^n$, $x_i \in R^d$, the goal of these methods is to specify a region for the target samples. The other well-known classifier with a different perspective is based on k-nearest neighbors (KNN), which decides if a sample belongs to the target class based on its distance to k-nearest data points in the training set, in contrast to SVDD and OCSVM that define a region for the target class. Thus, this classifier does not assume a fixed boundary for the target class, and the size of the boundary is flexible and reliant on the nearest neighbors of a test sample.

1) One-Class Support Vector Machine: The OCSVM aims to specify a function that takes a positive value for a small region that the training data live and take -1 elsewhere. This is done by maximizing the distance of the desired hyperplane to origin. The optimization problem is

$$\begin{aligned} &\min_{w,\psi,\rho} \ \frac{1}{2}\|w\|_2^2 + \frac{1}{\nu n}\sum_i \xi_i - \rho \\ &\text{s.t. } w^T\phi(x_i) \geq \rho - \xi_i, \quad i = 1, \ldots, n \\ &\quad \xi_i \geq 0 \end{aligned} \tag{3}$$

where $\phi(\cdot)$ is the high-dimensional space to which data are mapped and ν is the prior probability for the fraction of outliers. The OCSVM decision function can be computed using the kernel trick as

$$f(x) = \text{sign}(w^T\phi(x) - \rho) = \text{sign}\left(\sum_i \alpha_i K(x, x_i) - \rho\right) \tag{4}$$

where x is a test sample, $K(\cdot,\cdot)$ is the kernel function used for the training, *sign* is the sign function, and α is the Lagrangian multiplier of problem (3).

2) Support Vector Data Description: This method also seeks to estimate a region for the target class. In contrast to the OCSVM, the SVDD has a distinct approach to computing the desired region for the OCC. The SVDD describes the region of training samples as a hypersphere that is characterized by a center a and a radius R in the feature space. The corresponding minimization is

$$\begin{aligned} &\min_{R,a,\xi} \; R^2 + C\sum_i \xi_i \\ &\text{s.t. } \|\phi(x_i) - a\| \le R^2 + \xi \\ &\quad \xi \ge 0 \end{aligned} \tag{5}$$

where C is the regularization parameter. Given a new test sample z, it is from the desired class if its distance from the center is less than R.

3) K-Nearest Neighbors: KNN is another technique from the boundary-method class, but it does not seek a region for the target class. Instead, it classifies a test sample based on its distance to its nearest neighbors. In other words, this method does not assume a fixed region for the target class. Rather, the region is flexible and reliant on the nearest neighbors of the test sample. This algorithm has no training algorithm, and the decision for the test sample z is directly obtained based on the training samples. The underlying notion behind one-class KNN is the ratio between the distance of the sample z from its k neighbors and the distance of its k-neighbors from their nearest neighbors. In order words, assume that $q_i, i = 1, \ldots, k$ are the KNNs of the test sample z and $\hat{q}_i, i = 1, \ldots, k$ are the nearest target sample to q_i from the training data. The decision function $\rho(\cdot)$ of this method for the test sample z is

$$\rho(z) = \frac{\sum_{i=1}^{k} \|z - q_i\|}{\sum_{i=1}^{k} \|q_i - \hat{q}_i\|}. \tag{6}$$

If the value of $\rho(z)$ is less than a predefined threshold, then z is deemed to belong to the target class. Although the method has no training, the computation of (6) is time- and memory-costly since we need to hold all the samples in memory, and also, we have to compute the distance of a test sample to other points to get the KNNs.

C. Reconstruction Methods

The underlying idea of this class of methods is that the target-related test samples must be properly reconstructed from the training samples at hand. The proper construction is typically measured by the construction error, which is the distance between a test sample z with its reconstructed point $\hat{z}$. The difference between various methods of this class is basically their difference in constructing test samples based on the training data. The decision is also made simple; if the construction error of a test sample is less than a threshold, which is determined beforehand, then it belongs to the target class, and it is otherwise outlier. In the following, three well-known reconstruction-based one-class classifiers are discussed.

1) Principal Component Analysis: Principal component analysis (PCA) is a data-transformation technique that can be applied to cases where data lie on a linear subspace. The computation for the transformed data is based on the eigenvalue decomposition of the covariance matrix: The top $\hat{d}$ eigenvectors pertaining to the top $\hat{d}$ eigenvalues span a lower dimensional space that best represents the overall data.

For OCC, we transformed the target samples into a lower dimensional space by using PCA. Thus, the training $X \in R^{d\times n}$ is transformed into $\hat{X} \in R^{\hat{d}\times n}$, where $\hat{d} \ll d$. For a test sample z, we need to find its projection $\hat{z}$ into the transformed space $\hat{X}$ and compute the reconstruction error by finding the distance between z and $\hat{z}$. The projection of z onto the subspace is simply as

$$\hat{z} = \hat{X}(\hat{X}^T\hat{X})^{-1}\hat{X}^T z = \hat{X}^T\hat{X}z$$

and therefore, the reconstruction error for the test sample z is computed as

$$\epsilon(z) = \|z - \hat{z}\|^2 = \|z - \hat{X}^T\hat{X}z\|^2$$

where $\epsilon(z)$ is the reconstruction error for the test sample z. If this error is less than a threshold, the sample belongs to the target class.

2) K-Means: K-means is one of the first techniques in unsupervised learning, which aims to separate the input data into a predefined number of clusters, i.e., k. Each cluster is represented by its center; therefore, the outcome of k-means is k centers of clusters. For OCC, we first cluster training data into k groups and represent the center of each cluster by $\mu_i, i = 1, 2, \ldots, k$. Having k centers from k-means, the distance of a test sample z from each center is computed. If the distance of the sample z to only one of k centers is less than a threshold, then z is said to be a target sample. Otherwise, if the distances of the sample z to all cluster centers are larger than the given threshold, it is then an outlier.

3) Autoencoder: An autoencoder is a neural network that is well-known to learn the data representation. The autoencoder aims to reproduce the pattern at the input layer in the output layer. Therefore, the objective function for training this neural network is the distance between the input and output layers. For the OCC, the training samples of the target class are subjected to the autoencoder so that the weights of the neural network are computed. Then, for a test sample z, the output of the trained network for the input z is deemed as its reconstructed point $\hat{z}$, i.e., $\hat{z} = f_{\text{ae}}(z)$, where $f_{\text{ae}}(z)$ is the output of the trained autoencoder for the input z. The reconstruction error for the test sample z is then simply computed as

$$\epsilon(z) = \|z - f_{\text{ae}}(z)\|^2.$$

Similar to other methods in this class, if the reconstruction error is less than a predefined threshold, then the sample belongs to the target class; otherwise, it is an outlier.

V. Feature Extraction

In this section, the features used in this paper to represent tweets are explained. In total, 86 features are extracted from

each tweet. Features are classified into three categories: linguistic and content, user, and meta-message. The linguistic and content features capture synthetic and semantic aspects of the rumors, the features related to the users and their social networks are in the user category, and all the features about tweets metadata fall into the meta-message category. We deliberately skipped features related to the propagation and temporal aspects of the tweets as we wanted to identify rumors as early as possible, and those features are not available during the initial rumor propagation phase [1]. In Sections V-A–V-C, we go through each category of features and introduce them.

A. Linguistic and Content Features

The linguistic and content analysis enables us to scrutinize semantic and syntactic aspects of a tweet. For this research, we carry out this analysis by extracting features inspired by the existing literature of the rumor detection regarding both aspects. Table II lists all the linguistic and content features. The ones with a diamond (◇) show synthetic features, and the semantic features are indicated with box (□) symbol.

1) Syntactic Features: For the syntactic layer, some of the frequently used features are the number of words and characters and frequency of punctuation marks. Some context flavored features are the number of uppercase and lowercase characters and the number of capital words as some studies shown capital letters are the sign of rumor-prone tweet. There are other features regarding grammatical roles of the words in a sentence, such as frequency of first/second/third person pronouns and frequency of part-of-speech (POS) tags. POS explains what grammatical role a word plays in a sentence. Here, we consider 19 types of POS tags and count the frequency of every tag in each tweet.

There are three other syntactic features that are recently proposed by Vosoughi *et al.* [39]: number of abbreviations, average word complexity, and sentence complexity. The number of abbreviations counts how many abbreviations are used in a tweet. To count them, we first made a list, including 2622 abbreviations using online dictionaries [2] and Crystal's book on the language used on the Internet [40], and then check tweets against this list.[3] The average word complexity is estimated by the average length of words in a tweet. For example, the tweet *I just tried cooking popcorn with four mobile phones its a lie I tell you A LIE* has 17 words containing 1, 4, 5, 7, 7, 4, 1, 6, 6, 3, 1, 3, 1, 4, 3, 1, and 3 characters, respectively. The average word complexity is, therefore, 3.5. The sentence complexity of a tweet is estimated by the depth of its dependence parse tree. We used Stanford CoreNLP to generate dependence tree.

2) Semantic Features: To study the semantic layer, one of the most common approaches scholars adopt is capturing the emotions, attitude, or mood conveyed by a piece of text. For measuring such semantic proxies, in this paper, we borrow relevant features suggested in the literature, such as tone, subjectivity, polarity, and number of positive and negative words. There are also features inspired by the rumor literature, such as anxiety score, tentativeness score, opinion and insight score, certainty score, and number of vulgar words as many of the rumors emerge at the time of uncertainty and are not communicated with a decent and formal language.

For anxiety, tentativeness, opinion, and certainty scores, we used Linguistic Inquiry and Word Count (LIWC).[4] LIWC is a tool for analyzing the cognitive and psychological theme of text documents [41]. For vulgar words, we made a collection using online dictionaries,[5] including 1585 terms and checked each tweet against this collection.[6] The other features with high potential in capturing feelings are the number of emoticons and the number of emojis.[7] Both these pictographs indicate different types of feelings via a tiny figure which sometimes need many words to be described. The other feature is frequency of named-entity-recognition (NER) tags. NER is a technique to identify and segment the named entities and categorize them under various predefined classes (e.g., organization, place, and people). Here, we consider 17 types of NER tags and count the frequency of every tag in each tweet.

TABLE II

LIST OF FEATURES EXTRACTED FROM CONTENT OF TWEETS. FEATURES WITH DIAMOND (◇) AND BOX (□) SYMBOL ARE SYNTACTIC AND SEMANTIC FEATURES, RESPECTIVELY

List of linguistic & content features
◇ Number of question marks in a tweet [33], [42]
◇ Number of exclamation marks in a tweet [33], [42]
◇ Number of characters in a tweet [33]
◇ Number of words in a tweet
◇ Number of uppercase letters in a tweet [33], [42]
◇ Number of lowercase letters in a tweet [33]
◇ Number of capital words in a tweet
◇ Average word complexity in a tweet [39]
◇ Number of first person pronoun in a tweet [33], [43]
◇ Number of second person pronoun in a tweet [33]
◇ Number of third person pronoun in a tweet [33]
Number of vulgar words in a tweet [39]
◇ Number of abbreviations in a tweet [39]
□ Number of emoticons in a tweet [39]
□ Number of emojis in a tweet
□ Polarity of a tweet [5], [44]
□ Subjectivity of a tweet
□ Tone of a tweet
□ Positive words score of a tweet
□ Negative words score of a tweet
◇ Frequency of POS tags in a tweet [45]
□ Frequency of NER tags in a tweet
□ Opinion and insight score [39]
□ Anxiety score [31]
□ Tentativeness score [39]
□ Certainty score
□ Sentence complexity [39]

B. User Features

User analysis enables us to investigate the credibility of Twitter accounts. In this paper, we carry out this analysis via

[2]http://www.netlingo.com/category/acronyms.php

[3]The list of abbreviations is publicly available in this address: http://bit.ly/2Bxim4e.

[4]http://liwc.wpengine.com/compare-dictionaries/

[5]https://www.noswearing.com/dictionary and https://www.cs.cmu.edu/~biglou/resources/

[6]The list of vulgar terms is publicly available at http://bit.ly/2CtO4Rz

[7]The list of emoticons is publicly available at http://bit.ly/2EDDhG2.

TABLE III
FEATURES THAT WERE EXTRACTED FROM USER PROFILE

List of user features
Profile description (binary) [33], [5], [44], [46]
Verified account (binary) [33], [5], [46], [42]
Number of Statuses [5], [46], [33], [1]
Influence [5], [46], [33], [44], [1]
Number of following [5], [46], [44], [1]
User role
Total likes
Account age (day) [33]
Protected account (binary)
Profile location (binary)
Profile picture (binary)
Profile URL (binary)
Average follow speed
Average being followed speed
Average like speed
Average tweet speed
Screen name length
Number of digits in screen name

extracting features related to the account holders and their social network. Table III indicates the list of user features in this paper. Some of the simplest binary user features that are extensively alluded in literature are profile description, profile picture, profile URL, and verified account. These features are all about the account holder itself, while there are features regarding friendship network of account holders, such as number of followings, influence,[8] average follow speed,[9] average being followed speed, and user role. User role measures the ratio of followers to followees for a user. A user with a high follower-to-followee ratio is a broadcaster. Conversely, a user with a low follower-to-followee ratio is considered as a receiver. There are other features for user analysis regarding the user activity, such as number of statuses, total likes, average like speed, and average tweet speed. Moreover, two features are borrowed from social bot detection literature that deals with the screen name of the user: screen name length and number of digits in screen name.

C. Meta-Message Features

As with the linguistic and content analysis, in the meta-message feature category, a unit of analysis is tweet itself; however, here, the focus is on tweet metadata, not its textual content. Table IV demonstrates the list of meta-message features in this paper. We carry out the message analysis by extracting the basic message features that are frequently mentioned in the literature, such as number of hashtags and number of mentions. There are also some features regarding credibility of the message, such as tweet URL that checks whether a message contains a URL or not, number of multimedia that counts the number of multimedia (video or image) evidence accompanying the message, and geotagged tweet that checks whether the user is willing to left any trace about her location or not. Additionally, there

[8]In some studies, it is called the number of followers.

[9]All the average speed features are calculated by dividing the value of feature by the account age.

TABLE IV
FEATURES THAT WERE EXTRACTED FROM TWEET METADATA

List of meta-message features
Number of hashtags in a tweet [3]
Number of mentions in a tweet [33], [3]
Tweet URL (Binary) [33]
Number of multimedia in a tweet [5], [46], [44]
Number of likes [44]
Number of retweets [5], [46]
Tweet creation time (second)
Geotagged tweet (binary)

are features respecting the tweet exposure network: number of likes and number of retweets as whenever someone likes or retweets a tweet, the tweet appears in all of her followers' timeline.

VI. EXPERIMENT

In this section, we first explain the details of the experimental setup, then report the results of our experiments, and interpret them from three different perspectives.

A. Experimental Setup

As we discussed in Section III, designing an experiment for the binary classification and OCC is quite similar. Both need feature extraction, training and test data sets, and performance score to evaluate them. For training and test sets, in this paper, we use two renowned publicly available data sets (see Table V): Zubiaga [4] and Kwon [1] data sets. We refer to these data sets as the Zubiagaset and Kwonset, respectively. They are quite renowned in this field and have been appeared in several research studies [47]–[50]. They cover a wide variety of topics, including disaster, health, and politics, to name but a few. They are among the few publicly available data sets in this field.[10]

In the first data set, Zubiaga *et al.* [4] used Twitter Streaming API to collect tweets in two different situations: 1) breaking news that is likely to spark multiple rumors and 2) specific rumors that are identified *a priori*. Tweets are collected from five cases of breaking news. Given the large volume of tweets in the early stages of the data collection, they only sampled the tweets that provoked a high number of retweets. Then, they manually annotated the tweets as either rumor or non-rumor. In total, they collected 6425 tweets, including 4023 non-rumor and 2402 rumor tweets.

In the second data set, Kwon *et al.* [1] made a list of popular rumors by searching fact-checking websites. They also made another list for non-rumors by searching for notable events from news media outlets. Then, they crawled one of the largest and near-complete repositories of Twitter to collect tweets relevant to the list of rumors and non-rumors. In total, they identified 140 910 tweets for 111 events (44 394 tweets from 60 rumors and 96 516 tweets from 51 non-rumors). To the best of our knowledge, Kwon data set is the biggest publicly available data set in this field to date.

[10]There are other large data sets in this field [2], [39], but they are either not publicly available [39] or cannot be used for any purposes except for the validation of the original study [2].

TABLE V
STATISTICAL INFORMATION REGARDING ZUBIAGA [4] AND KWON [1] DATA SETS

	Zubiagaset	Kwonset
Number of tweets	6,425	140,910
Number of rumours	2,402	44,394
Number of non-rumour	4,023	96,516
Description	Based on five events	Based on 111 events

After preparing the data sets, the next step is feature extraction in which every tweet is represented by 86 features. Then, the one-class classifier must be trained. This is the only distinct step of OCC compared to the binary classification. Unlike the binary classification that needs both classes for training, we train the one-class classifier with one class only. In the rumor detection problem, this means that training a binary classifier needs both rumor and non-rumor while one-class classifier is trained only by rumor. To test the performance of one-class classifiers, we follow the same evaluation approach as the binary classification and test the classifier with both rumor and non-rumor (see Table VI).

To get more reliable results on the data sets, we use k-fold cross validation. In this technique, the rumor data set is partitioned into k bins of equal size, and then, we perform k separate learning experiments by picking $k-1$ folds of rumor data set for the training and one remaining fold along with non-rumor class for the test in each experiment. In the end, the average performance of the k experiments is reported as the performance of the model. In this paper, we repeat k-fold cross validation for $k = 3, 5, 10$ to show the sensitivity of the models' performance to the training sample size.

For the classifier selection, we consider seven classifiers belonging to three OCC paradigms, including Gaussian classifier as a density estimation method, one-class support vector machine (OCSVM), SVDD, and KNN as boundary methods, and K-means, PCA, and autoencoder as reconstruction methods. For parameters tuning and sensitivity analysis over the model hyper-parameters, we use a grid search technique and measure the models' performance regarding the different combinations of hyper-parameters. In kernel-based methods, namely, SVDD and OCSVM, we used the radial basis function (RBF) as it was suggested and used in many of rumor detection works [6]. To apply the selected algorithms, we used MATLAB and Python. The OCSVM is implemented in scikit-learn (a python machine learning library) [51], and the rest of the algorithms come with MATLAB PRTools [52] package.

From the implementation point of view, in the above-mentioned programming libraries and packages, the methods that are essentially using kernel or distance matrix are not well suited for relatively large data sets. We tried to tackle this issue by establishing a powerful computer system to perform the experiments; however, it could not manage to run SVDD over Kwonset. Therefore, we decided to perform SVDD experiments by subsampling the Kwonset.

We report the model performance via precision, recall, and F1-score. Precision is the fraction of correctly retrieved instances that are relevant, while recall is the fraction of relevant documents that are retrieved. F1-score is the harmonic mean of precision and recall [53]. We also assess the models' efficiency by measuring the execution time of the experiments in both data sets.

B. Results and Discussion

In this section, we report and discuss the results of the experiments. To this end, we first make a baseline analysis by comparing the results of the experiments with the baseline of each data set. Then, we measure the impact of training sample size on the models' performance. After that, we evaluate the impact of hyper-parameters in each model. Then, we report the models' performance in different feature categories, and finally, we assess the models' execution time in each data set.

1) Baseline Analysis: In this section, we study the performance of one-class classifiers in comparison with baselines in both operational data sets. In the first baseline, Zubiaga *et al.* [4] proposed a rumor identification classifier using CRF with fivefold cross validation, and in the second one, Kwon *et al.* [1] applied a random forest with threefold cross validation. Both baselines use F1-score, precision, and recall to report classifiers' performance. For the baseline analysis, we replicate the same conditions as the original studies; therefore, for the Zubiagaset and Kwonset, we perform the experiments in fivefold and threefold cross validation, respectively. We also report the classifiers' performance with the same metrics as the original studies.

Table VII demonstrates the precision, recall, and F1-score of classifiers in both data sets along with the results of baselines. Based on Table VII, in the Zubiagaset, KNN, k-means, and SVDD outperform baseline with the F1-score of 74.30%, 64.73%, and 63.54%, respectively. In the Kwonset, all seven one-class classifiers achieve better F1-score than the baseline and thus outperform it.

In terms of precision and recall, in the Zubiagaset baseline, precision is better than recall. This is another way around for the one-class classifiers on the same data set, which means that recall is higher than precision in one-class classifiers. On the other hand, in the Kwonset, baseline precision and recall are almost equal, while precision is slightly higher than recall in one-class classifiers.

Compared to baseline, in the Zubiagaset, one-class classifiers can identify more rumors but with less precision. Therefore, it is highly likely to identify the bigger fraction of rumors as well as mislabeling the bigger fraction of non-rumors as rumor when we use one-class classifiers in the Zubiagaset. Repeating the same comparison in the Kwonset results in the same number of identified rumors by one-class classifiers but with higher precision. This means that it is highly likely to identify the same fraction of rumors but with less mistaken flags when we use one-class classifiers in Kwonset.

We can infer that one-class classifiers often outperform baselines' binary classifiers or achieve to their close proximity in spite of the fact that one-class classifiers are trained in the absence of non-rumor samples.

2) Training Sample Size Impact: In this section, we analyze the impact of training sample size on the performance of the

TABLE VI
CONFUSION MATRIX FOR OCC [12]

	Object from target class	Object from outlier class
Classified as a target object	True positive, T^+	False positive, F^+
Classified as an outlier object	False negative, F^-	True negative, T^-

TABLE VII
BASELINE ANALYSIS ON THE ZUBIAGASET AND KWONSET [1], [4]. WE COULD NOT APPLY SVDD ON THE WHOLE KWONSET SINCE THE STANDARD SOLVER OF SVDD DOES NOT SUIT THE LARGE-SCALE DATA SETS. WE TACKLED THIS PROBLEM BY SUBSAMPLING THE TRAINING SET AND EXPERIMENT WITH A SUBSET OF THE ORIGINAL DATA SET

	Zubiagaset			Kwonset		
	PR	RE	F1	PR	RE	F1
Autoencoder	48.61%	**77.90%**	59.86%	**97.31%**	**90.14%**	**93.59%**
Gaussian	38.33%	**87.80%**	53.36%	**95.69%**	90.03%	**92.77%**
K-means	53.82%	**81.20%**	**64.73%**	**96.11%**	90.08%	**93.00%**
KNN	**69.20%**	**80.20%**	**74.30%**	**98.19%**	90.11%	**93.98%**
SVDD	51.78%	**82.20%**	**63.54%**	**95.20%**	**91.21%**	**93.16%**
OCSVM	13.08%	51.30%	20.85%	**95.99%**	88.24%	**91.95%**
PCA	41.38%	**90.20%**	56.73%	**96.99%**	90.03%	**93.38%**
Baseline	66.7%	55.6%	60.7%	89%	90%	89.5%

classifiers. To this end, we measure the performance of different one-class algorithms with k-fold cross validation when $k \in \{3, 5, 10\}$. Since F1-score combines precision and recall, we use it to represent classifiers performance. Fig. 6 shows how changing the training sample size affects classifiers' performance. It constitutes of two subfigures that represent the performance changes in each data set correspondingly.

By comparing classifiers' performance with different cross validations, we can observe two different patterns: performance gain and performance loss by the growth of the training sample size. In the Zubiagaset, except SVDD and OCSVM, the other classifiers' performance improves as the training sample size increases. In the Kwonset, all the classifiers but autoencoder and SVDD experience performance enhancement when the training sample size grows.

Despite the heterogeneous impact of training sample size on the classifiers' performance, the difference between the highest and lowest performances in 10 out of 14 classifiers is less than 2%. This means that our models show a high level of robustness against training sample size alteration.

3) Hyper-Parameters Impact: In this section, we discuss the sensitivity of the models to the hyper-parameters value. We look at the classifiers hyper-parameters in each of the data sets and between them. Table VIII summarizes the classifiers hyper-parameters and their valid range. Each classifier has two hyper-parameters which we change them within their valid range.

Fig. 7 shows the classifiers F1-score regarding different combinations of hyper-parameters. For each classifier, two heatmaps are used to represent the performance space in both Zubiagaset and Kwonset. In the heatmaps, darker colors represent higher performance.

As shown in Fig. 7, for each classifier (in both data sets), the best performance is achieved when the target-class error is in the vicinity of its lowest value. Some of the classifiers, such as Gaussian and KNN, are indifferent to the second hyper-parameter, which for instance in the case of KNN, it does not matter how many neighbors are selected; however, for some others, it is not the case and the best performance depends on both hyper-parameters. For example, in OCSVM, the best performance is achieved when the kernel parameter has a small value or autoencoder performs better when the number of hidden units is higher. By an inter-data set analysis, we can see that classifiers pursue a similar hyper-parameters' pattern in both data sets. This means that the high and low levels of performance for a classifier are achieved in similar areas in hyper-parameter space. We have also observed that SVDD is almost indifferent to its hyper-parameters and perform similarly in different combinations of hyper-parameters.

4) Classifiers Performance: In this section, we report and discuss the results of the experiments in different feature categories. Fig. 8 shows the performance metrics of the one-class classifiers on two data sets for different feature categories. Fig. 8 consists of two panels representing the performance scores in different data sets. Each panel is also composed of three parts that display the performance of different feature categories in terms of F1-score, precision, and recall. We first report classifiers' performance in the Zubiagaset where all features are considered. Then, we go through Kwonset and report one-class classifiers performance when all features are present. Finally, we investigate the synergy between the feature categories in both data sets.

As shown in Fig. 8 (left), among the one-class classifiers that are trained on the Zubiagaset with full features, KNN has the highest F1-score. It also has the highest precision and lowest recall among the same group of classifiers. SVDD has the second-highest F1-score and precision, while it outperforms other classifiers. Although SVDD has the best recall and second best precision, KNN's high precision compensates its low score in recall and gives it the highest F1-score. After KNN and SVDD, the next one-class classifiers with the highest F1-score are K-means, autoencoder, PCA, and Gaussian. The lowest performance belongs to OCSVM, which delivers the poor F1-score of 28% by 81% recall and 17% precision.

The experiments on the Kwonset are also shown in Fig. 8 (right). In this set of experiments, we could not apply SVDD

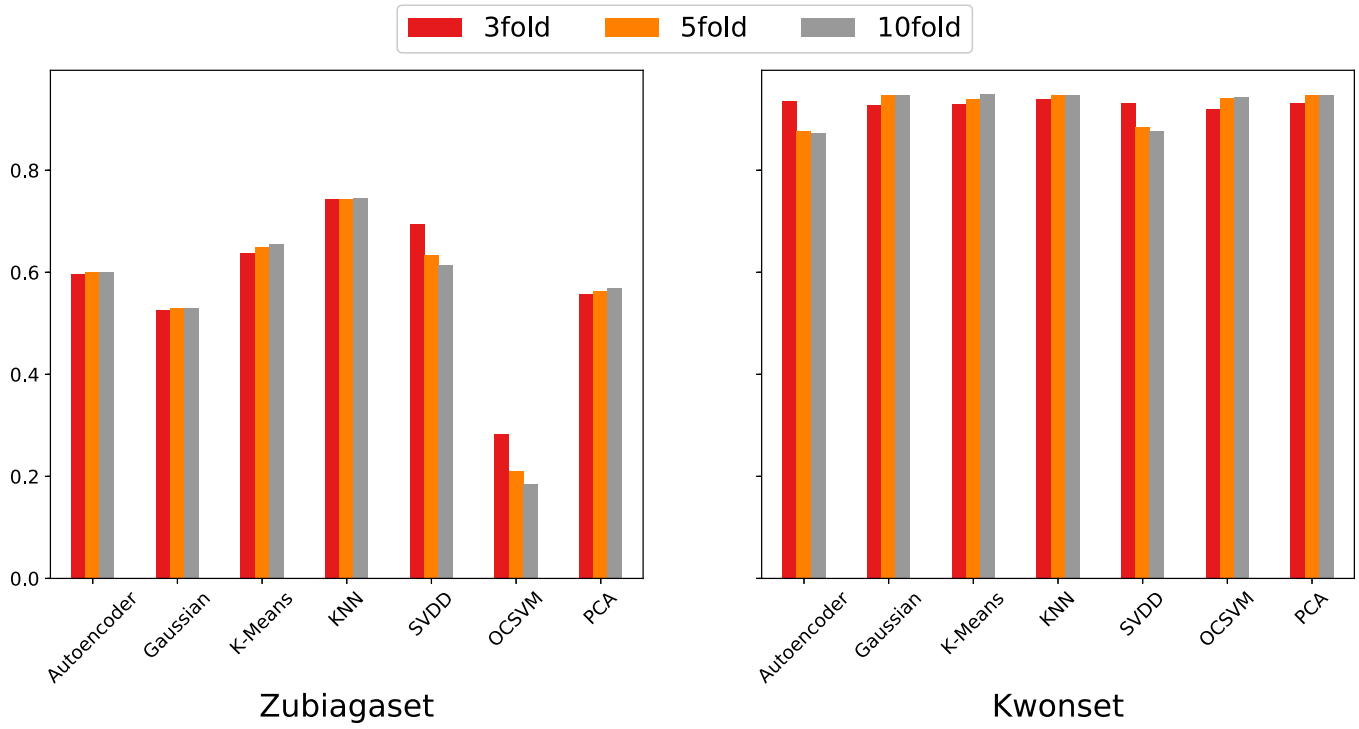

Fig. 6. Impact of training sample size on the performance of classifiers in the Zubiagaset and Kwonset.

TABLE VIII
CLASSIFIERS HYPER-PARAMETERS AND THEIR VALID RANGE.

Classifiers	Hyper-parameters	Valid range
Autoencoder	FRACREJ: Fraction of target objects rejected N: Number of hidden units	FRACREJ $\in (0,1)$ $\{i \mid i \in N, i \leq datapoints\}$
Gaussian	FRACREJ: Error on the target class R: Regularization parameter	FRACREJ $\in (0,1)$ R $\in (0,1)$
K-means	FRACREJ: Error on the target class K: Number of clusters	FRACREJ $\in (0,1)$ $\{i \mid i \in N, i \leq datapoints\}$
KNN	FRACREJ: Error on the target class K: Number of neighbors	FRACREJ $\in (0,1)$ $\{i \mid i \in N, i \leq datapoints\}$ FRACREJ $\in (0,1)$
SVDD	FRACREJ: Error on the target class P: Inverted kernel parameter	FRACREJ $\in (0,1)$
OCSVM	ν: Regularization parameter γ: Kernel parameter	$\nu \in (0,1)$
PCA	FRACREJ: Error on the target class N: Number of PCA components	FRACREJ $\in (0,1)$ $\{i \mid i \in N, i \leq features_number\}$

on the whole Kwonset since the standard solver of SVDD does not suit the large-scale data sets. We tackled this problem by subsampling the training set and conduct the experiment on a subset of the original data set. For the rest of the classifiers, despite the long execution time, the experiments were finished and produced expected outcomes. In the Kownset, all full features trained classifiers achieve a high level of F1-score that is obtained by a high level of precision and recall. The performance of one-class classifiers is very close to each other, in such a way that the difference between maximum and minimum of F1-score, precision, and recall is less than 1%. The high level of both precision and recall means that one-class classifiers could identify most of the rumors with a high level of precision.

Last but not least, by considering Fig. 8, one can simply realize that the synergy of all features from the three categories is positive based on F1-score. In particular, the F1-score of the autoencoder, Gaussian, *K*-means, KNN, SVDD, OCSVM, and PCA based on all features is superior to the F1-score obtained by training with individual feature categories only. However, this superiority is more significant in some classifiers, for instance, SVDD in the Zubiagaset or Gaussian in Kwonset. In some of the experiments, such as the KNN trained by the linguistic features on the Zubiagaset, recall is higher compared to the training with all features, but its precision is also lower, which results in the overall lower F1-score.

5) Classification Speed: The other metric to assess and compare the classifiers is their speed. To measure the classification speed, we gauge the execution time of classification, which means the average time of training and test for one iteration of k-fold cross validation. Fig. 9 shows the execution time of classifiers across the Kwonset and Zubiagaset. We use

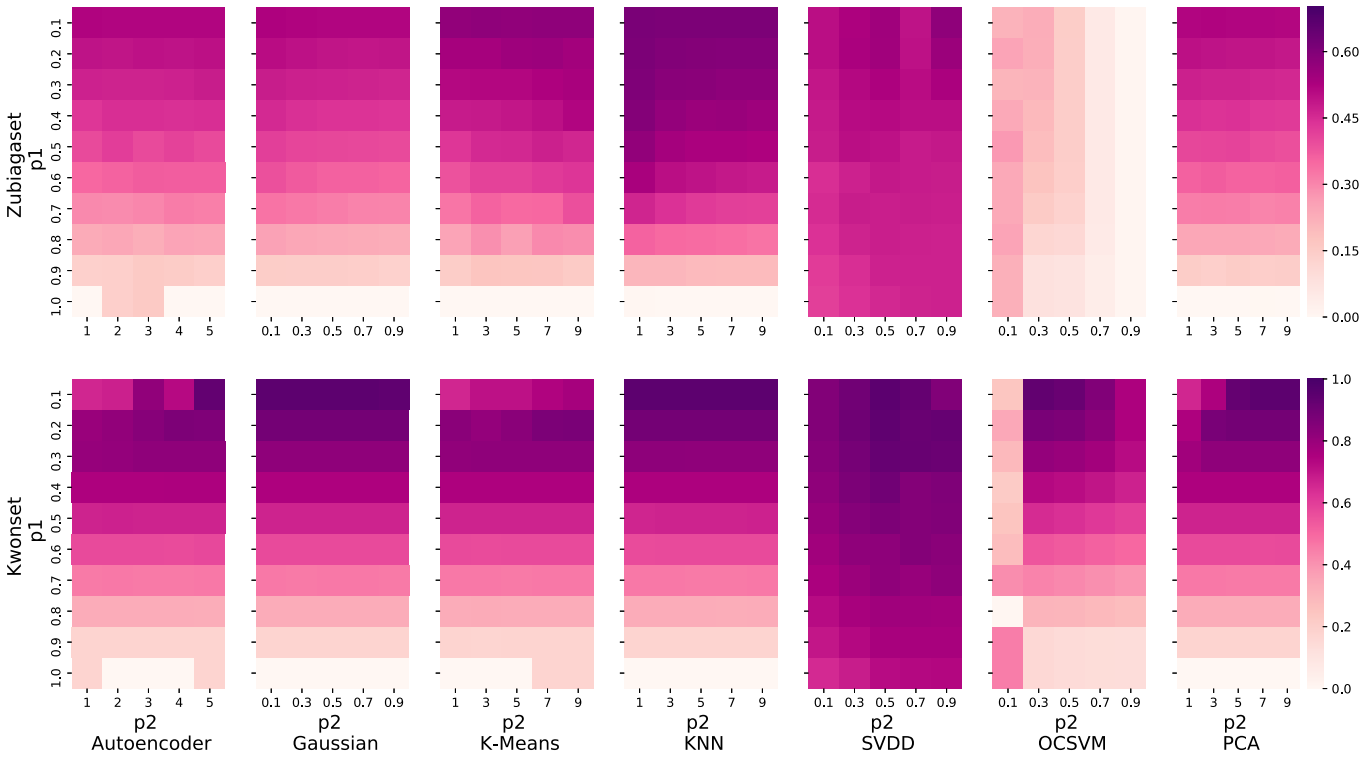

Fig. 7. Impact of hyper-parameters on models performance in the Zubiagaset and Kwonset.

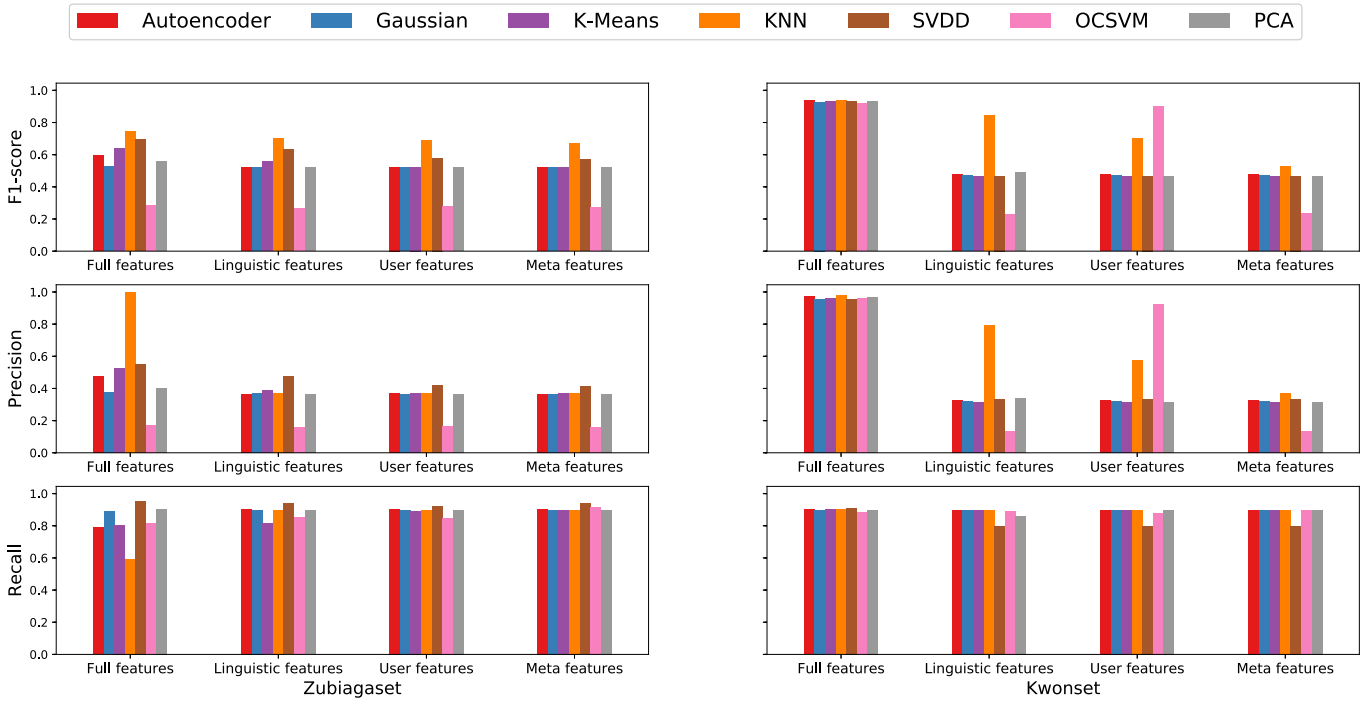

Fig. 8. Classifiers' performance in different feature categories in the Zubiagaset and Kwonset.

a log–log diagram due to the substantial difference between the execution times.

In both data sets, Gaussian, K-means, and PCA are the fastest classifiers. The execution time of the other classifiers varies in both data sets. In the Zubiagaset, SVDD is the slowest model, while in Kwonset, KNN has the longest execution time. It is worth mentioning that due to the scalability problem in the MATLAB SVDD package, we use a subset of Kwonset for SVDD. In the Zubiagaset, after SVDD, autoencoder, KNN, and OCSVM are the slowest classifiers. In Kwonset, the next three classifiers with the longest execution time are OCSVM, autoencoder, and SVDD.

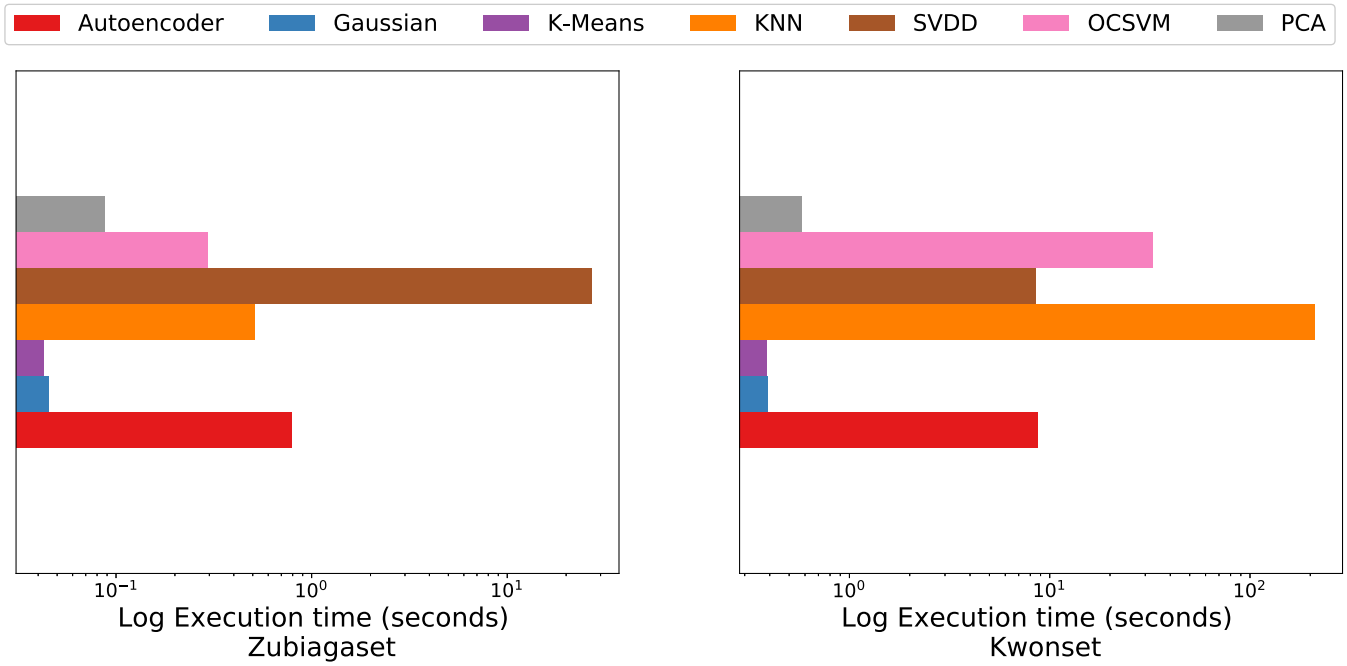

Fig. 9. Execution time of classifiers in the Zubiagaset and Kwonset.

VII. Conclusion

In this paper, we studied the binary rumor classification pitfall by addressing the long-standing and unnoticed concept of "non-rumor." Non-rumor is a term coined by computer scientists when they formulated rumor detection as a binary classification problem. There is neither a clear definition nor a consensus among scholars for this pseudo-concept. Some studies imply that rumor and non-rumor are complementary concepts; in other words, non-rumor is any piece of information that cannot get rumor label. In some other work, non-rumor is considered as factual information. This ambiguity and lack of consensus about the definition of non-rumor has influenced data annotation, which eventually diminished the quality of classification. It also impacts the data annotation in a way that a tweet may take the non-rumor label as it is a piece of information from a credible news source, while in another research, a tweet may take the same label because it is not a rumor.

This lack of consensus on data annotation prevents us from making a comparison between different classifiers as they do not annotate (consciously or unconsciously) tweets consistently. It also makes the classifiers unreliable as there is no unanimous correct definition for non-rumor, and we do not know which one is the right definition. Based on the stated flaws, we reach to the conclusion that the binary classification with the current approach to non-rumor might not be beneficial to the computational rumor detection.

To tackle the issue addressed earlier and avoid dealing with non-rumor, we adopted a novel classification approach called OCC. This approach goes very well with the special characteristics of our problem as the classifier is trained by one class only. Hence, it provides us with an opportunity to have a classifier for detecting rumor without touching the controversial area of non-rumor. For the feature extraction, we took two principles into account, first, to focus on early available features due to the importance of early detection of the rumors and, second, to consider features that have already been suggested as effective features for rumor detection in social networks.

To evaluate the quality of the proposed approach, we trained seven one-class classifiers on two major data sets and compared their performance. We observed that the OCC approach can recognize rumors with a high level of F1-score. In the Zubiagaset, this approach could achieve the F1-score of 74%, and in Kwonset F1-score, this approach reaches 93%. We extended the experiments to different feature categories and analyzed the performance of each classifier on individual feature categories. We observed a positive synergy when individual feature categories are aggregated. We also studied the impact of training sample size and hyper-parameters on the classifiers' performance. We reported the model performance in different settings using F1-score, precision, and recall. Additionally, to understand the efficiency of the one-class classifiers, we compared their speed by measuring the execution time of each classifier.

We can summarize our findings into a few lessons learned. SVDD performs very well in terms of precision and recall; however, it is not time and memory efficient. Hence, it is an ideal one-class classifier for small-sized problems. KNN performs very well in terms of precision, while its performance subjected to recall is poor. It is also time and memory inefficient but not as bad as SVDD. If the data set is not too large and precision is the main concern, KNN can be the right choice. The other one-class classifier is autoencoder, which achieves good results with respect to both precision and recall. In contrast to SVDD and KNN, it is memory efficient and can

be executed on a desktop computer even for relatively large data sets, such as Kwonset. On the other hand, it is time-consuming, especially when it is a setup with a high number of hidden units. Therefore, in the case of relatively large data sets, lack of adequate computational resources, having a considerable amount of time autoencoder can be a proper one-class classifier. Gaussian, K-means, and PCA produce mediocre results with a relatively low level of precision and a high level of recall in small data sets; however, they are very fast even in large data sets. Therefore, if the data sets are large, they are ideal choices.

For the future works, several directions can be built upon this research. The first one is using more data points for training the classifier and studying how increasing data points can affect the performance of the rumor classifiers. However, due to the expensive procedure of data collection and annotation (time- and money-wise), a possible way of data generation would be transfer learning and generative adversarial networks (GANs). Another avenue for future research would be a methodological improvement. We observed that OCSVM performs poorly and excellently, respectively, on the Zubiagaset and Kwonset, while the rest of the classifiers show an acceptable level of performance in the Zubiagaset. Evidently, some OCC algorithms perform brilliantly in some data sets and poorly on some others. A possible research suggestion can be applying an ensemble OCC technique for rumor identification. Another methodological avenue can be developing an efficient solver for the SVDD technique as it crashes when the size of the data grows. The current solver of SVDD is developed by the creators of this method and is based on MATLAB Optimization Toolbox. The fourth research direction can be extracting more features influenced by the literature of rumor psychology as well as features regarding the context of the rumor.

REFERENCES

[1] S. Kwon, M. Cha, and K. Jung, "Rumor detection over varying time windows," *PLoS ONE*, vol. 12, no. 1, Jan. 2017, Art. no. e0168344.

[2] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online.," *Science*, vol. 359, no. 6380, pp. 1146–1151, Mar. 2018.

[3] G. Liang, W. He, C. Xu, L. Chen, and J. Zeng, "Rumor identification in microblogging systems based on users' behavior," *IEEE Trans. Computat. Social Syst.*, vol. 2, no. 3, pp. 99–108, Sep. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7419281/

[4] A. Zubiaga, M. Liakata, and R. Procter, "Learning reporting dynamics during breaking news for rumour detection in social media," Oct. 2016, *arXiv:1610.07363*. [Online]. Available: https://arxiv.org/abs/1610.07363

[5] F. Yang, Y. Liu, X. Yu, and M. Yang, "Automatic detection of rumor on Sina Weibo," in *Proc. ACM SIGKDD Workshop Mining Data Semantics*, 2012, Art. no. 13.

[6] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter, "Detection and resolution of rumours in social media: A survey," *ACM Comput. Surv.*, vol. 51, no. 2, Jun. 2018, Art. no. 32.

[7] R. Sicilia, S. Lo Giudice, Y. Pei, M. Pechenizkiy, and P. Soda, "Twitter rumour detection in the health domain," *Expert Syst. Appl.*, vol. 110, pp. 33–40, Nov. 2018.

[8] A. Zubiaga, M. Liakata, R. Procter, K. Bontcheva, and P. Tolmie, "Crowdsourcing the annotation of rumourous conversations in social media," in *Proc. 24th Int. Conf. World Wide Web*, New York, NY, USA, 2015, pp. 347–353

[9] A. Zubiaga, M. Liakata, R. Procter, G. W. S. Hoi, and P. Tolmie, "Analysing how people orient to and spread rumours in social media by looking at conversational threads," *PLoS ONE*, vol. 11, no. 3, Mar. 2016, Art. no. e0150989.

[10] J. H. M. Janssens, "Outlier selection and one-class classification," Ph.D. dissertation, Tilburg Centre Cognition Commun., Tilburg Univ., Tilburg, The Netherlands, 2013.

[11] S. S. Khan and M. G. Madden, "One-class classification: Taxonomy of study and review of techniques," *Knowl. Eng. Rev.*, vol. 29, no. 3, pp. 345–374, Jan. 2014.

[12] D. M. J. Tax, "One-class classification: Concept-learning in the absence of counter-examples," Ph.D. dissertation, Dept. Med. Phys. Biophys., Delft Univ. Technol., Delft, The Netherlands, 2001.

[13] N. DiFonzo, P. Bordia, and R. L. Rosnow, "Reining in rumors," *Org. Dyn.*, vol. 23, no. 1, pp. 47–62, 1994.

[14] W. A. Peterson and N. P. Gist, "Rumor and public opinion," *Amer. J. Sociology*, vol. 57, no. 2, pp. 159–167, Sep. 1951.

[15] J. Prasad, "The psychology of rumour: A study relating to the great Indian earthquake of 1934," *Brit. J. Psychol. Gen. Sect.*, vol. 26, no. 1, pp. 1–15, 1935.

[16] R. H. Knapp, "A pyschology of rumour," *Public Opinion Quart.*, vol. 8, no. 1, pp. 22–37, 1944.

[17] G. W. Allport and L. Postman, *The Psychology of Rumor*, 1st ed. New York, NY, USA: Henry Holt and Company, 1947.

[18] H. T. Buckner, "A theory of rumor transmission," *Public Opinion Quart.*, vol. 29, no. 1, pp. 54–70, 1965.

[19] N. DiFonzo, *The Watercooler Effect: An Indispensable Guide to Understanding and Harnessing the Power of Rumors*. New York, NY, USA: Avery, 2009.

[20] N. Difonzo and P. Bordia, "Rumor in organizational contexts," in *Advances in Social and Organizational Psychology*, D. A. Hantula, Ed. Abingdon, U.K.: Routledge, 2006, ch. 13, pp. 249–274.

[21] T. A. Knopf, *Rumors, Race and Riots*. Piscataway, NJ, USA: Transaction, 1975.

[22] F. Koenig, *Rumor in the Marketplace: The Social Psychology of Commercial Hearsay*. Auburn, MA, USA: Auburn House, 1985.

[23] N. DiFonzo and P. Bordia, *Rumor Psychology: Social And Organizational Approaches*. Rosie Phillips Davis: American Psychological Association, 2007.

[24] A. Louni and K. P. Subbalakshmi, "Who spread that rumor: Finding the source of information in large online social networks with probabilistically varying internode relationship strengths," *IEEE Trans. Comput. Social Syst.*, vol. 5, no. 2, pp. 335–343, Jun. 2018.

[25] G. Tong, W. Wu, and D.-Z. Du, "Distributed rumor blocking with multiple positive cascades," *IEEE Trans. Computat. Social Syst.*, vol. 5, no. 2, pp. 468–480, Jun. 2018.

[26] D. Brockmann and D. Helbing, "The hidden geometry of complex, network-driven contagion phenomena," *Science*, vol. 342, no. 6164, pp. 1337–1342, Dec. 2013.

[27] X. Ren, N. Gleinig, D. Helbing, and N. Antulov-Fantulin, "Generalized network dismantling," *Proc. Nat. Acad. Sci. USA*, vol. 116, no. 14, pp. 6554–6559, Apr. 2019.

[28] D. J. Daley and D. G. Kendall, "Epidemics and rumours," *Nature*, vol. 204, no. 225, p. 1118, Dec. 1964.

[29] L. Zhu, G. Guan, and Y. Li, "Nonlinear dynamical analysis and control strategies of a network-based SIS epidemic model with time delay," *Appl. Math. Model.*, vol. 70, pp. 512–531, Jun. 2019.

[30] X. Yang, Y. Wu, J. Zhang, and T. Zhou, "Dynamical behavior of rumor spreading under a temporal random control strategy," *J. Stat. Mech., Theory Exp.*, vol. 2019, no. 3, Mar. 2019, Art. no. 033402.

[31] N. Turenne, "The rumour spectrum," *PLoS ONE*, vol. 13, no. 1, Jan. 2018, Art. no. e0189080.

[32] S. Kwon, M. Cha, K. Jung, W. Chen, and Y. Wang, "Prominent features of rumor propagation in online social media," in *Proc. IEEE 13th Int. Conf. Data Mining*, Dec. 2013, pp. 1103–1108.

[33] C. Castillo, M. Mendoza, and B. Poblete, "Information credibility on Twitter," in *Proc. 20th Int. Conf. World Wide Web (WWW)*, New York, New York, USA, 2011, pp. 675–684.

[34] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, and K. F. Wong, "Detecting rumors from microblogs with recurrent neural networks," in *Proc. IJCAI*, 2016, pp. 3818–3824.

[35] D. A. Bird, "Rumor as folklore: An interpretation and inventory," Ph.D. dissertation, Univ. Indiana, Bloomington, IN, USA, 1979.

[36] D. A. Bird, S. C. Holder, and D. Sears, "Walrus is greek for corpse: Rumor and the death of paul McCartney," *J. Popular Culture*, vol. 10, no. 1, pp. 110–121, Jun. 1976.

[37] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 582–588.

[38] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, Jan. 2004.

[39] S. Vosoughi, M. N. Mohsenvand, and D. Roy, "Rumor gauge: Predicting the veracity of rumors on Twitter," *ACM Trans. Knowl. Discovery Data*, vol. 11, no. 4, pp. 1–36, Jul. 2017.

[40] D. Crystal, *Language and the Internet*. Cambridge, U.K.: Cambridge Univ. Press, 2006.
[41] J. W. Pennebaker, M. R. Mehl, and K. G. Niederhoffer, "Psychological aspects of natural language use: Our words, our selves," *Annu. Rev. Psychol.*, vol. 54, no. 1, pp. 547–577, Feb. 2003.
[42] A. Zubiaga, M. Liakata, and R. Procter, "Exploiting context for rumour detection in social media," in *Proc. Int. Conf. Social Inform.* Oxford, U.K.: Springer, 2017, pp. 109–123.
[43] J. Ma, W. Gao, Z. Wei, Y. Lu, and K. Wong, "Detect rumors using time series of social context information on microblogging websites," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, New York, NY, USA, 2015, pp. 1751–1754.
[44] Q. Zhang, S. Zhang, J. Dong, and J. Xiong, "Automatic detection of rumor on social network," in *Natural Language Processing and Chinese Computing*. Beijing, China, 2015, pp. 113–122.
[45] O. Varol, C. A. Davis, F. Menczer, and A. Flammini, *Feature Engineering for Social Bot Detection*. Boca Raton, FL, USA: CRC Press, 2018, pp. 311–334.
[46] K. Wu, S. Yang, and K. Q. Zhu, "False rumors detection on Sina Weibo by propagation structures," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 651–662.
[47] J. Kim, B. Tabibian, A. Oh, B. Schölkopf, and M. Gomez-Rodriguez, "Leveraging the crowd to detect and reduce the spread of fake news and misinformation," in *Proc. 11th ACM Int. Conf. Web Search Data Mining (WSDM)*, New York, NY, USA, 2018, pp. 324–332.
[48] S. A. Alkhodair, S. H. H. Ding, B. C. M. Fung, and J. Liu, "Detecting breaking news rumors of emerging topics in social media," *Inf. Process. Manage.*, Feb. 2019.
[49] O. Ajao, D. Bhowmik, and S. Zargari, "Fake news identification on Twitter with hybrid CNN and RNN models," in *Proc. 9th Int. Conf. Social Media Soc. (SMSoc)*, New York, NY, USA, 2018, pp. 226–230.
[50] S. Das Bhattacharjee, A. Talukder, and B. V. Balantrapu, "Active learning based news veracity detection with feature weighting and deep-shallow fusion," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2017, pp. 556–565.
[51] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[52] R. P. W. Duin *et al.*, "PRTools: A MATLAB toolbox for pattern recognition," Delft Univ. Technol., Delft, The Netherlands, Tech. Rep., 2000, pp. 109–111.
[53] C. D. Manning, P. Raghavan, and H. Schutze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.
[54] R. L. Rosnow, "Inside rumor: A personal journey," *Amer. Psychol.*, vol. 46, no. 5, p. 484, 1991.

Amir Ebrahimi Fard received the B.Sc. degree in software engineering from the Iran University of Science and Technology (IUST), Tehran, Iran, and the M.Sc. degree in business and economics from the Sharif University of Technology (SUT), Tehran. In his Ph.D. studies, he is part of the Engineering Social Technologies for a Responsible Digital Future Project at TU Delft, where he is involved in novel approaches to identify rumors in online social networks.

He is currently a Ph.D. Researcher with the Policy Analysis Section, Faculty of Technology, Policy and Management, Delft University of Technology (TU Delft), Delft, The Netherlands. His current research interests include computational social science, machine learning, and social network analysis.

Majid Mohammadi received the B.Sc. degree in software engineering and the M.Sc. degree in artificial intelligence from the Ferdowsi University of Mashhad, Mashhad, Iran. He is currently pursuing the Ph.D. degree with the Information and Communication Technology Group, Department of Technology, Policy and Management, Delft University of Technology, Delft, The Netherlands.

His current research interests include semantic interoperability, machine learning, and pattern recognition.

Yang Chen (M'07–SM'15) received the B.S. and Ph.D. degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2004 and 2009, respectively.

He visited Stanford University, Stanford, CA, USA, in 2007, and Microsoft Research Asia, Beijing, from 2006 to 2008 as a Visiting Student. From April 2011 to September 2014, he was a Post-Doctoral Associate with the Department of Computer Science, Duke University, Durham, NC, USA, where he served as Senior Personnel for the NSF MobilityFirst Project. From September 2009 to April 2011, he was a Research Associate and the Deputy Head of the Computer Networks Group, Institute of Computer Science, University of Göttingen, Göttingen, Germany. He is currently an Associate Professor with the School of Computer Science, Fudan University, Shanghai, China, where he leads the Mobile Systems and Networking (MSN) Group. His current research interests include online social networks, Internet architecture, and mobile computing.

Dr. Chen has served as an OC/TPC Member for many international conferences, including SOSP, WWW, IJCAI, AAAI, IWQoS, ICCCN, GLOBECOM, and ICC. He serves as an Associate Editor for the IEEE ACCESS and an Editorial Board Member of the *Transactions on Emerging Telecommunications Technologies* (ETT).

Bartel Van de Walle is currently a Full Professor with the Chair of Policy Analysis, Faculty of Technology, Policy and Management, Delft University of Technology, Delft, The Netherlands. He is also the Head of the Department of Multi Actor Systems, Delft University of Technology. His current research interests include the role of (information) technology for decision-making for individuals and groups in complex situations, in particular in the response to humanitarian crises.

Prof. Van de Walle was a Senior Fellow at the Harvard Humanitarian Initiative and the Harvard Kennedy School's Crisis Leadership Program in the Ash Center for Democratic Governance and Innovation for the period 2015–2017. He is also a member of the Agder Academy of Sciences and Letters, Norway.