Published paper available at: http://dx.doi.org/10.1016/j.inffus.2022.11.029

SAIRUS: Spatially-Aware Identification of Risky Users in Social Networks

Antonio Pellicani[a], Gianvito Pio[a,b,*], Domenico Redavid[a], Michelangelo Ceci[a,b,c]

[a] *Dept. of Computer Science, University of Bari, Via Orabona, 4, 70125 Bari, Italy*
[b] *Big Data Lab, National Interuniversity Consortium for Informatics (CINI), Via Volturno, 58, 00185 Roma, Italy*
[c] *Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia*

Abstract

The massive spread of social networks provided a plethora of new possibilities to communicate and interact worldwide. On the other hand, they introduced some negative phenomena related to social media addictions, as well as additional tools for cyberbullying and cyberterrorism activities. Therefore, monitoring operations on the posted contents and on the users behavior has become essential to guarantee a safe and correct use of the network. This task is even more challenging in presence of borderline users, namely users who appear risky according to their posts, but not according to other perspectives.

In this context, this paper contributes towards an automated identification of risky users in social networks. Specifically, we propose a novel system, called SAIRUS, that solves node classification tasks in social networks by exploiting and combining the information conveyed by three different perspectives: the semantics of the textual content generated by users, the network of user relationships, and the users spatial closeness, derived from the geo-tagging data associated with the posted contents. Contrary to existing approaches that typically inject features built from one perspective into the other, we learn three separate models that exploit the peculiarity of each kind of data, and then learn a model to fuse their contribution using a stacked generalization approach.

Our extensive experimental evaluation, performed on two variants of a real-world Twitter dataset, revealed the superiority of the proposed method, in comparison with 13 competitors based on one of the considered perspectives alone, or on a combination thereof. Such a superiority is also clear when specifically focusing on borderline users, confirming the applicability of SAIRUS in real-world social networks, which are potentially affected by noisy data.

*Corresponding author
Email addresses: antonio.pellicani@uniba.it (Antonio Pellicani), gianvito.pio@uniba.it (Gianvito Pio), domenico.redavid1@uniba.it (Domenico Redavid), michelangelo.ceci@uniba.it (Michelangelo Ceci)

Keywords: Social Network Analysis, Node Classification, User Risk Identification, Spatial Analysis

1. Introduction

In the globalized world we live in, social networks play a central role in connecting people, due to the possibility to share news about our lives and to express our opinions. Indeed, by performing common actions such as writing a post, adding a *like* to comments and photos, or following the updates of influencers, users can establish new relationships, share ideas, beliefs, and preferences, or discuss about specific topics and events.

The ubiquity of social networks inspired the scientific community, which over time analyzed several aspects of this phenomenon. In particular, Social Network Analysis (SNA) processes have been widely used to exploit the relationships and the information flows among users in the network [1]. Using SNA approaches, social networks may be exploited for several goals, ranging from advertising interesting products to specific users [2], to understanding the political debate of voters and their polarization near the elections [3, 4]. In this context, our goal is to analyze social networks to identify the so-called *risky* users, namely users who exploit the spreading power of social networks to perform and incite bad or illegal activities, including the use of drugs, the embracement of religious or political extremism, and the hate towards women or disabled people [5, 6, 7]. The identification of risky users may therefore be fundamental to promptly suspend suspicious accounts and stop such activities [8, 9, 10, 11].

From a methodological viewpoint, the identification of risky users can be framed as a node classification task. Multiple approaches have been proposed in the literature to solve node classification tasks, that mainly fall into three main categories: content-based approaches, topology-based approaches and hybrid approaches. The first category relies on the analysis of the content generated by users [12, 13, 14, 15]. Conversely, topology-based approaches take into account only the relationships between the users [16, 17, 18, 19]. A possible relationship in the network may represent, for example, a user who follows the updates/posts of another users, a user who likes the content shared by another users, or a user who comments a post shared by another user.

In the context of the identification of risky users, both the approaches may encounter issues in the classification of *borderline users*. A typical example of such users is represented by journalists. Indeed, they may usually publish posts containing *unsafe* words, increasing the chance of misclassifications for content-based approaches. Similarly, they may establish mixed relationships with both safe and risky users. In such a scenario, if the relationships with the safe users are not predominant, topology-based approaches would erroneously classify journalists as risky users. Solving these issues is the goal of hybrid approaches [20, 21], that try to combine the approaches falling in the first two categories to exploit their strengths and possibly alleviate their weaknesses.

It is noteworthy that the massive adoption of social networks is also due to the possibility to interact with them using mobile devices (i.e., smartphones and tablets). Most of the mobile devices integrate geolocation mechanisms, based on GPS sensors, accelerometers, and magnetometers. When a new post or image is shared, provided that the necessary permissions have been granted by the user, additional personal information are linked to the content loaded on the social network, thus generating geotagged data. However, to the best of our knowledge, existing approaches are not able to consider the information possibly conveyed by the geographical position of the users, that implicitly establish additional relationships among them.

In this paper, we aim to fill this gap. Specifically, we propose SAIRUS, a hybrid user risk identification framework, capable to consider not only the content generated by the users and their relationships in the network, but also the spatial dimension through their geographical position. The goal is to possibly improve the performance of the learned node classification models and the robustness to the presence of borderline users, by exploiting the spatial closeness among users.

SAIRUS learns three different node classification models (one for each perspective to consider), which are finally fused to get the final, possibly more robust, user risk classification model, based on the stacked generalization framework [22]. As regards the content, we learn a word embedding model and exploit the embedded content to train two autoencoders specialized in recognizing safe and risky users, respectively. As regards the user relationships and spatial closeness, we extract two separate embeddings representing topological and spatial information, and train two different classifiers on top of the learned representations. Contrary to existing hybrid approaches that are usually based on the injection of artificially-defined features related to one perspective into the others [23, 24, 25], the approach adopted by SAIRUS allows us to focus separately on the three different perspectives and learn a final classifier that ultimately combines their contribution.

The remaining of the paper is organized as follows: in Section 2 we briefly discuss some related work; in Section 3 we describe the details of the proposed framework; in Section 4 we describe the results of our experimental evaluation; finally, in Section 5 we draw some conclusions and outline possible future works.

2. Background

A social network is commonly seen as a *virtual square* where users share their thoughts and ideas, even if the concept of social network existed long before the massive diffusion of Web 2.0. The first studies about Social Network Analysis (SNA) stem from sociology [26] and aim to analyze social relationships between people. Starting from the 1990s, it has been applied to several fields including Physics, Political Science, Biology, Psychology, or Economics. SNA is strongly coupled with graph theory, through which it abstracts the human relationships using *nodes* and *links*. Specifically, each node in the network represents an actor, i.e., a person or an organization, while links represent social relationships between the actors [27].

Nowadays, SNA is exploited to support many scenarios, including critical situations in the context of the homeland security. Some relevant examples include the analysis of the spread of the COVID-19 pandemic [28], the investigation of the mechanisms that trigger macro-level international migration patterns [29, 30], the analysis of the euroscepticism in the British Parliament before the vote for the Brexit [31], or the prediction of Bin-Laden's replacement as head of al-Qaeda [32]. SNA can also be applied in the counter-terrorism field. For example, in [33] the authors stated that SNA can be considered a powerful tool for the analysis of terrorist and criminal networks, since it can effectively be adopted to support many crucial tasks including *key-player identification* [34, 35] and *link analysis* [36, 37].

As already mentioned in Section 1, the goal of the proposed method SAIRUS is to identify risky users, i.e., users who negative influence the community through their actions in the social network. This kind of task falls in the *key-player identification* category and can be practically solved by resorting to *node classification* approaches. For this reason, in Section 2.1 we briefly discuss existing methods aiming to solve user classification tasks in social networks. Moreover, since our method specifically exploits the spatial dimension, in Section 2.2 we introduce some existing spatially-aware approaches that generally work on network data.

2.1. User classification in social networks

The general goal of the user classification task in social networks is to assign a category or a label to each user. As mentioned in Section 1, existing methods can be categorized in three main categories: content-based, topology-based, and hybrid approaches, depending on the type of information they use.

A relevant example of content-based methods can be found in [38], where the authors propose a method that exploits sentiment analysis. Starting from tweets, a NLP preprocessing pipeline is applied and a sentiment score is calculated for each meaningful word. Finally, a decision tree is learned to assign a category to each user between *positive*, *neutral*, and *negative*. In the context of content-based methods, the adoption of Word2Vec [39] and Doc2Vec [15] is also very common. Both methods allow to learn an embedding numerical space, where each textual document is represented. They have a different granularity: Word2Vec naturally returns a numerical vector for each word, bringing out latent semantic meanings and relationships among words (such as synonymy or polysemy); on the other hand, Doc2Vec focuses on entire paragraphs (or documents). In both cases, the obtained numerical representation of the text can be subsequently used for any downstream task, such as classification. Relevant examples of works exploiting this pipeline can be found in [12, 40, 41].

Considering the specific case of detecting risky users, Hee *et al.* [42] focused on the detection of cyberbullying content in social media texts. In particular, their system can recognize blasphemies or defamation. After a NLP-based preprocessing phase, they extract vectors of features from the tweets exploiting *n-gram bag-of-words* and topic modeling algorithms like *Latent Dirichlet Allocation* [43]. In the last step they learn a linear support vector machine classifier, that shows good results on English and Dutch datasets. In a similar context, in [14] authors combined classical weighting schemes, like TF-IDF or binary weighting, with fuzzy sets, creating a fuzzy set-based weighting method for the detection of cyber terror and extremist content.

Focusing on topology-based methods, we can mention the system *GNetMine* [16], a graph-based transductive classification approach, which can also model heterogeneous information networks consisting of multiple types of nodes and links. Other topology-based methods solve node classification tasks by resorting to *collective inference* [44, 45, 46], which consists in taking concurrent decisions on the label of every nodes, rather than classifying each node separately. Due to its nature, if a collective inference model is trained on a noisy dataset (i.e., a dataset containing weak or wrong relationships), misclassification of unlabeled nodes are propagated to nodes in their neighborhood, generating a *domino effect*. Focusing on this issue, in [47] the authors proposed an active inference method capable to identify a portion of the misclassified network and correct the label of nodes, improving the classification results. Analogously, in [48] the authors proposed to weight the relationships between existing nodes by counting the number of connections through each of them. The authors showed that this approach is particularly useful to avoid *weak relationships* from influencing the final node classification. Finally, it is worth mentioning the work in [18], where the authors solved a within-network classification task on a partially-labeled network. This is a challenging scenario, in which relational learning is combined with semi-supervised learning to enhance the classification performance in a sparse network. In particular, the authors exploited the so-called *ghost edges*, i.e., artificially-introduced edges between every labeled node and the unlabeled node to classify. Each ghost edge is weighted with a proximity score, calculated exploiting random walks with restart. Finally, labels are propagated through the ghost edges, taking into account the calculated weights.

As for hybrid methods, a first attempt to combine both content and network-derived information was proposed in [49], where the authors analyzed a network composed by nodes representing users and hashtags to learn a classifier that is able to distinguish between verified and unverified users. A link between two nodes represents the fact that a user mentions another user, or uses an hashtag. The authors proposed to build a set of features from both the constructed network and the textual content, that is subsequently exploited to train a decision tree. However, the adopted representation is not able to take into account typical relationships of social networks, such as *friends* or *followers*.

In [50], a framework to automatically recognize rebel users in social networks was presented. The authors combined features extracted from both the content and the user profile, along with features extracted from a semantic user graph constructed over the content. The graph transforms the tweets into a structure similar to an ontology. Here, the semantics of the tweets is made explicit through the links connecting the subject word with the object word, traversing the verb.

It is important to note that the mentioned hybrid methods combine the features from both the user profile and the user-generated content, possibly adopting a network/graph as a proxy, but they do not analyze directly the network structure established by the relationships among users.

Among more complex approaches, naturally able to work with heterogeneous attributed networks, it is worth mentioning the system *HENPC* [51] which is able to classify multi-type nodes exploiting overlapping and hierarchically organized clusters. In the same line of research, *MrSBC* [52] and its ensemble-based variant *MT-MrSBC* [53], consider both the attributes and the relationships between the nodes, exploiting the naïve Bayes classification method in the multirelational network setting. Contrary to [49] and [50], the methods [51, 52, 53] are more tailored for the analysis of the network structure and, although each node can be associated to attribute values, these methods are not able to explicitly consider the semantics of the textual content. Specifically, in heterogeneous networks, nodes might represent users, posts, or single words. Therefore, the user-generated content is represented through the relationships between nodes of type *user* and nodes of type *words*, without the possibility of modeling the semantics represented by the words or sequences of words.

In this context, and contrary to all the mentioned approaches, SAIRUS is able to analyze both the semantics of the content and the topology of the network in which the user is involved. Furthermore, SAIRUS explicitly considers the spatial closeness among the users involved in the network, allowing to classify them more accurately. As far as we know, no existing method has considered the spatial dimension, together with the user-generated content and relationships, when accomplishing this task.

2.2. Spatially-aware methods for network data

The spread of spatial and geo-referenced data renewed and incentivized the interest towards Geographic Information Systems (GIS) [54], spatial data analysis [55] and spatial data mining [56]. The latter refers to the process of discovering useful and previously unknown patterns from spatial databases [57, 58]. Due to the exploitation of the spatial dimension of social networks, this paper also contributes to these fields. In the following, we briefly discuss existing methods that attempted to consider the spatial dimension.

In [59] the authors identify spatial regions with a higher risk of infection by Dengue disease, training two probabilistic models from Twitter data generated by users located in two Brazilian cities. The considered data include tweets and users GPS positions, through which the authors detected individuals who had a personal experience with the disease. Then, they reconstructed the position history of each user to identify spatial clusters and to highlight those with a higher infection risk.

Similar approaches have been proposed to detect risky spatial clusters based on criminal events [60] or traffic accidents [61], although they do not exploit spatial data extracted from social networks. In [62], the authors proved the effectiveness of a multidimensional analysis when investigating the spread of extreme weather events, like El Niño. They analyzed the risk perception of the storm reaching the US West Coast, showing that considering the spatial dimension could help in answering questions about the climate changes, and to provide insights about discussions on Twitter.

In the literature, we can also find some attempts to exploit spatial data for the detection of terrorists on social networks. A very simple approach is proposed in [50], where the authors use the presence of geo-tagged tweets as a feature, assuming that malicious users are less likely to share their location to remain hidden. Although simple and interesting, this approach, due to its basic assumptions, may lead to an excessive amount of false positives. Among more complex approaches, in [63] the authors performed an analysis of a terrorist social network, focusing on the geographical information which could be exploited to provide insights into the structure and the dynamics of the network. In particular, they show how this kind of data could help in identifying terrorist operational cells, along with their bases and their support facilities.

In summary, although few preliminary attempts have been made to exploit the spatial dimension in the analysis of (social) network data, the method SAIRUS presented in this paper can be considered the first that explicitly models the spatial relationships among the users to possibly improve the key-player identification task, and specifically the classification of users as *risky* or *safe*.

3. The proposed method SAIRUS

Before explaining in details the approach followed by the proposed method SAIRUS, we first formalize some key aspects. First, we formalize a social network as a 4-tuple as follows:

$$\langle N, C, E_C, E_T \rangle \tag{1}$$

where:

- $N = N_L \cup N_U$ $(N_L \cap N_U = \emptyset)$ is the set of users, either labeled (N_L) or unlabeled (N_U). Each labeled user is associated with the category *safe* or *risky*, thus implicitly defining two subsets of labeled users $N_L^{(s)}$ and $N_L^{(r)}$, such that $N_L = N_L^{(s)} \cup N_L^{(r)} (N_L^{(s)} \cap N_L^{(r)} = \emptyset)$.
- C is the set of textual documents produced by users, that is, the posts. Each document $c \in C$ is associated with a *timestamp* and a *geographical location*.
- $E_C \subseteq N \times C$ represents the relationships between users and textual contents, i.e., the action performed by a user in generating/posting a given textual content.
- $E_T \subseteq N \times N$ represents the topology of the network established by a possible social relationship between users, e.g., *follows*.

It is worth noting that, based on the data available during the training phase, the task of node classification in network data can be solved in two different settings: *(semi-supervised) inductive* setting [64] or *semi-supervised transductive* [65] setting. The former, also know as *across-network classification*, takes advantage of a model learned from a (fully or partially) labeled network to classify nodes in an unseen, unlabeled network. In the latter setting, which is also know as *within-network classification*, the model is learned from a network containing both labeled and unlabeled nodes, and the goal is to classify specifically the set of unlabeled nodes observed at training time, which is the common situation in social networks. Since SAIRUS classifies users in N_U, based on information learned from the whole set of users in N, it naturally falls into the category of *semi-supervised transductive* learning approaches.

We want to stress that SAIRUS solves this classification task by exploiting not only the topology of the network established by the relationships between labeled and unlabeled users, but also the textual content of their posts and the spatial closeness among users estimated on the basis of the locations associated with their posts.

As depicted in Figure 1, SAIRUS consists of four main stages: *i)* semantic content analysis of the textual documents produced by users, *ii)* topology network analysis on the user relationships, *iii)* analysis of the spatial closeness among users, *iv)* model fusion. SAIRUS exploits a stacked generalization approach to "learn to combine" the contribution coming from all the considered perspectives. On the contrary, as pointed out in Section 2.1, existing hybrid methods exhibit relevant limitations, such as: *i)* they exploit only few (and weak) spatial features (see [63]), or they do not consider the spatial dimension at all; *ii)* they only take into account simple topological features (see [49, 66]); or *iii)* user relationships are totally discarded (see [50]).

In the following subsections, we briefly describe each of the main stages performed by the components of SAIRUS.

3.1. Semantic analysis of the textual content

The aim of this component is to analyze the textual content (e.g., posts, tweets, comments, etc.) generated by users, and categorize unlabeled users as *safe* or *risky* accordingly. The input of this component consists of the set of textual documents C and the set of relationships E_C representing the link between users and the textual documents they posted/published. SAIRUS first applies some common Natural Language Processing (NLP) pre-processing steps [67] on the textual documents, namely *tokenization, stopword removal* and *stemming.* Subsequently, for each user, SAIRUS concatenates all the pre-processed documents posted by such a user. Note that the temporal order of the initial textual documents is considered during the concatenation, implicitly allowing SAIRUS to take into account the temporal evolution of the topics discussed by the user. This is an important aspect since the behavior of the users can be subject to a drift over time.

Before training a classifier, we need to represent users according to the textual content they posted, namely, as feature vectors representing the semantics of the textual content in a latent feature space. For this purpose, we adopt the well-established word-embedding method Word2Vec [39]. Specifically,

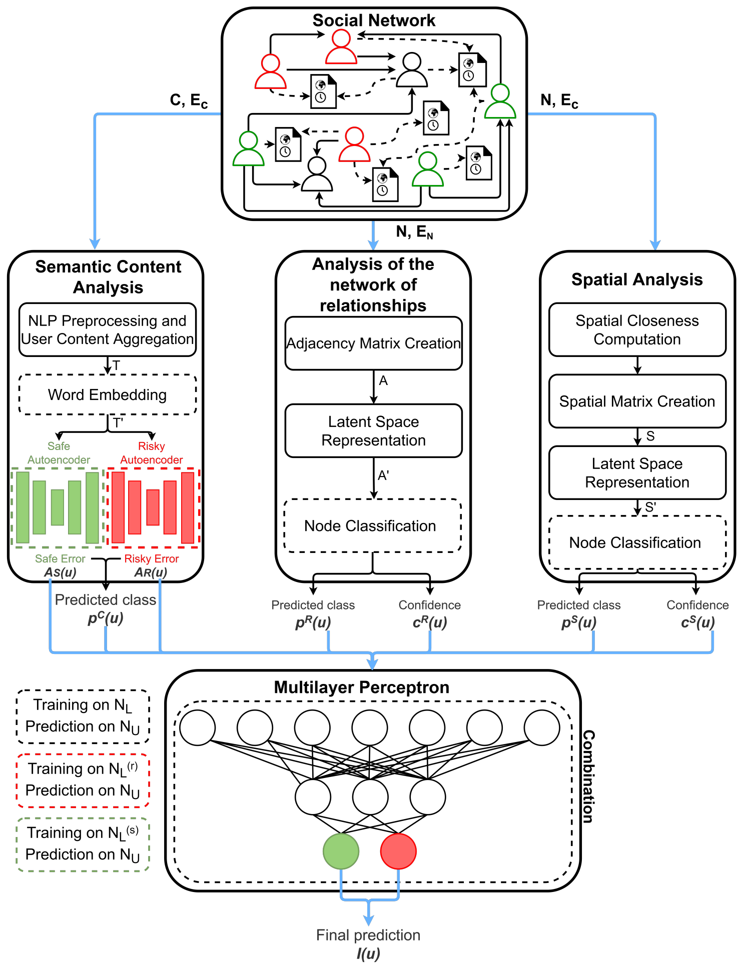

Figure 1: On overview of the SAIRUS architecture.

Word2Vec is able to represent each single word as a k_c-dimensional real-valued vector. To compute an embedding associated with the user, we rely on the *additive compositionality* property of word embeddings [68], which states that the meaning of the words can be composed by adding up their embeddings. More formally, given $words(u)$, the list of words appearing in the textual content posted by the user u, and $w2v(w)$, the embedding generated by Word2Vec for the word w, then the semantic vector representation $sem(u)$ for each user $u \in N$ is calculated as:

$$sem(u) = \sum_{w \in words(u)} w2v(w). \tag{2}$$

In SAIRUS, other word and document embedding techniques may be plugged in, such as BERT [69]. However, we decided to avoid the adoption of BERT in SAIRUS due to its limitations in processing sequences of words longer than 512 tokens, that can be easily reached in our scenario. Through the word embedding phase, we obtain a new dataset $T' \in \mathbb{R}^{|N| \times k_c}$, which consists of the semantics-based k_c-dimensional feature vector representation for all the users N.

In order to properly learn a classification model from such a dataset, we recall that users N can be labeled as *risky* ($N_L^{(r)}$), labeled as *safe* ($N_L^{(s)}$) or unlabeled (N_U). In this phase, we focus only on labeled users and learn two different one-class classifiers (one for each class) based on stacked autoencoders [70]. Autoencoders are popular neural networks exhibiting a funnel-shaped structure, that aims to learn a latent representation of the data such that the input is accurately reconstructed in the output layer. They exhibit state-of-the-art performances in classification tasks based on textual content [71, 72], being able to catch the semantics from the latent learned space, and have been successfully been applied also in anomaly detection [73, 74] and embedding [75, 76] tasks.

Formally, each autoencoder aims at learning an encoding function $\tilde{en} : \mathcal{X} \rightarrow \mathcal{X}'$ and a decoding function $\tilde{dc} : \mathcal{X}' \rightarrow \mathcal{X}$, such that:

$$\langle \tilde{en}, \tilde{dc} \rangle = \underset{\langle en, dc \rangle}{\operatorname{argmin}} \|T' - dc(ec(T'))\|^2, \tag{3}$$

where $\mathcal{X}$ is the input space (i.e., $\mathbb{R}^{k_c}$), and $\mathcal{X}'$ is the learned encoding space.

The architecture of an autoencoder consists of two main parts, associated with the encoding and the decoding stages, that are fully connected feedforward neural networks, with the same number of hidden layers arranged so that their architectures are mirrored. The central layer, called *embedding or bottleneck* layer, has an arbitrary dimension, usually smaller than the input layer, and represents the embedding space. Figure 2 shows the autoencoder architecture adopted in SAIRUS, with two hidden layers for each part.

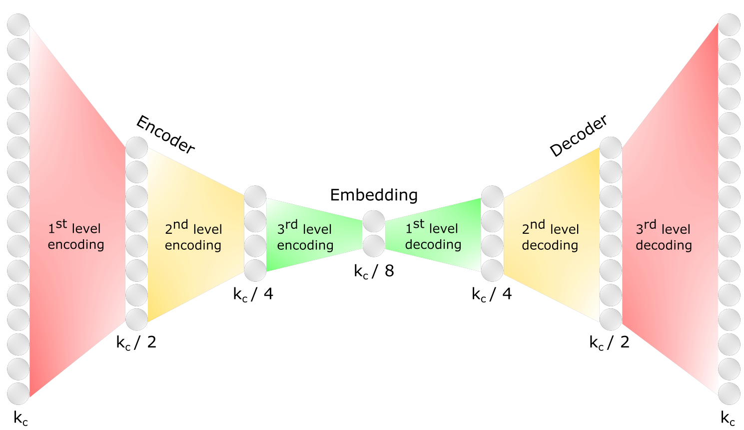

Figure 2: A graphical representation of the autoencoder architecture adopted in SAIRUS for semantic content analysis: three encoding stages and three decoding stages, that aggregate and reconstruct, respectively, the semantic representation of each user.

As previously mentioned, we build two separate autoencoders, one for each category of users. More formally, we train the autoencoder AR from the vector representation of labeled risky users $N_L^{(r)}$, and the autoencoder AS from the vector representation of labeled safe users $N_L^{(s)}$. Given an unlabeled user $u \in N_U$, we feed both the autoencoders AS and AR with his/her corresponding vector representation $sem(u)$, and compute the respective reconstruction errors $AS(u)$ and $AR(u)$. Therefore, the output of the semantic analysis of the textual content for a user $u \in N_U$ can be considered threefold:

- the reconstruction error $AS(u)$ achieved by the autoencoder AS;
- the reconstruction error $AR(u)$ achieved by the autoencoder AR;
- the predicted label $p^c \in \{S, R\}$ (safe or risky), computed according to the minimum error achieved by AS and AR.

We stress the fact that this component specifically focuses on the semantic analysis of the textual content. On the other hand, the aspects related to the topology of the network of relationships are captured by a specific component, that will be described in the following subsection.

3.2. Analysis of the network of relationships

The most straightforward approach to take into account the network of relationships among users consists in the analysis of an adjacency matrix $A \in \mathbb{R}^{|N| \times |N|}$, where $A_{ij} = 1$ if $(u_i, u_j) \in E_N$, $A_{ij} = 0$ otherwise, and u_i and u_j are the i-th and the j-th user of the network, respectively. However, the direct analysis of adjacency matrices through machine learning approaches usually suffers from issues arising from high dimensionality and sparseness. This is due to the large number of users in a social network, each of which naturally establishes relationships only with a few other users. For example, looking at the Facebook Results Report for Second Quarter 2021[1], the social network had over 1.9 billion of active daily users in June 2021, which would led to an adjacency matrix with over 3×10^{18} cells. Considering the maximum allowed number of friendships (5000), this matrix would be very sparse (sparsity $> 99.999\%$).

This is a well-known issue in the literature and there are many solutions relying on dimensionality reduction techniques, including *Singular Value Decomposition* (SVD) [77], *Principal Component Analysis* (PCA) [78] and *Non-negative Matrix Factorization* (NMF) [79], which solve the problem of low-rank matrix approximation, dealing with sparse data and facilitating the exploitation of latent information. There are also other approaches based on autoencoders [80], or that do not work on the adjacency matrix, but rather on the network itself. A relevant example is *Node2Vec* [81], which exploits random walks and the word embedding method *Word2Vec*, to construct node embeddings of a predefined

[1] https://investor.fb.com/investor-news/press-release-details/2021/Facebook-Reports-Second-Quarter-2021-Results/

dimension. It has also been proved that these methods support the modeling of communities and of the roles of the users in the community.

SAIRUS is able to work directly on the adjacency matrix $A \in \mathbb{R}^{|N|\times|N|}$, or on the resulting matrix $A' \in \mathbb{R}^{|N|\times k_r}$ obtained by the application of the PCA, autoencoder or Node2Vec, where k_r is a user-defined parameter. Note that any additional dimensionality reduction technique may be easily pluggable in the SAIRUS workflow. Subsequently, SAIRUS trains a node classification model from the whole set of labeled users N_L. In this case, although the classifier is trained from labeled users only, their embedding is constructed also considering their relationships with unlabeled users. This is coherent with the transductive semi-supervised learning setting.

For this phase we adopt tree-based classifiers because they generally exhibit state-of-the-art performances on classification tasks in the semi-supervised setting [82], also from network data [64]. Tree-based models are predictive models that are well known for their interpretability, their ability to handle both numerical and categorical data, as well as to capture non-linearities. Often exploited in multi-class classification scenarios, the learned decision trees consist of nodes and branches. They are usually learned through top-down induction approaches, that recursively partition the set of observations. Each node considers a specific feature and a value/threshold, according to which the observations are partitioned. In the leaf nodes, we can find the predicted labels (for classification tasks) or numerical values (for regression tasks). Each split is greedily determined by maximizing some heuristics. In particular, the decision tree learned by SAIRUS maximizes the reduction of the classical *Gini Index* [83], that is based on the purity of each class after applying the split. In our case, the Gini Index is defined as $Gini(n) = 1 - (p_s^2 + p_r^2)$, where p_s and p_r are the relative frequencies of safe and risky users in the tree node n, respectively.

During the prediction phase, given an unlabeled user $u \in N_U$, the decision tree built by SAIRUS provides the predicted label $p^R(u)$ and a confidence value $c^R(u)$. The confidence value associated with a given unlabeled user is based on the purity, computed on the training examples associated with the leaf node in which u falls. Both the predicted label $p^R(u)$ and the confidence value $c^R(u)$ are then exploited in the model fusion phase (see Figure 1).

3.3. Spatial analysis

In this subsection, we describe the approach we adopt to specifically take into account the spatial dimension. We first build a network represented as a weighted adjacency matrix $S \in \mathbb{R}^{|N|\times|N|}$, where $S_{ij} = closeness(u_i, u_j)$ corresponds to the spatial closeness between the user u_i and the user u_j. The function $closeness(u_i, u_j)$ is computed by exploiting the geodetic distance $d(u_i, u_j)$ between the geographical locations of the user u_i and of the user u_j. We approximate the geographical location of a given user as the mode of the geographical locations associated to his/her posts. The adoption of the mode, instead of other aggregation functions (such as the centroid) is motivated by its capability of *i)* associating the most relevant position to the user, discarding sporadic changes due to occasional travels; *ii)* returning a location in which such a user has really been located, rather than a synthetic *average* position which potentially may not represent a real possible location.

More formally, the geodetic distance relies on the Law of Haversines [84], that can determine the distance between two points on a sphere, given their latitudes and longitudes. Therefore, given two users u_1, u_2, their latitudes φ_1, φ_2 and their longitudes λ_1, λ_2, $d(u_i, u_j)$ is computed as:

$$d(u_i, u_j) = 2r \cdot \arctan \frac{\sqrt{a}}{\sqrt{1-a}} \tag{4}$$

where r is the average earth radius ($r \approx 6{,}371$ km) and $a = \sin^2(\frac{\varphi_2 - \varphi_1}{2}) + cos(\varphi_1) \cdot cos(\varphi_2) \cdot \sin^2(\frac{\lambda_2 - \lambda_1}{2})$ is the Haversine Formula.
Subsequently, we standardize the distance $d(u_i, u_j)$, using the z-score normalization, as follows:

$$z(u_i, u_j) = \frac{d(u_i, u_j) - \mu_d}{\sigma_d} \tag{5}$$

where μ_d and σ_d are the mean and the standard deviation, respectively, of the distances between two users.

We adopt z-standardization since it allows us to easily identify two main groups: users who are spatially closer than the average (i.e., $z(u_i, u_j) < 0$), and users who are spatially more distant than the average (i.e., $z(u_i, u_j) \geq 0$). Since we explicitly interested in representing the spatial closeness among users, we compute $closeness(u_i, u_j)$ as follows:

$$closeness(u_i, u_j) = \begin{cases} \dfrac{z(u_i, u_j)}{min_z}, & \text{if } z(u_i, u_j) < 0 \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where min_z is the minimum of the normalized distances between two users.

Note that we further normalize $z(u_i, u_j)$ over min_z in order to obtain a value in the range $[0, 1]$, where 0 means that the users u_i and u_j are very far from each other (actually, more than the average) and 1 means that u_i and u_j are located precisely at the same location.

Once the matrix S has been computed, analogously to the approach followed for the analysis of the network of relationships, we apply a dimensionality reduction technique, obtaining the reduced matrix $S' \in \mathbb{R}^{|N| \times k_s}$, where k_s is a user-defined parameter. Finally, we train a node classification model on the labeled users N_L. Coherently with the case of the network of relationships, also in this case, we adopt a decision tree learner based on the Gini index, that, given an unlabeled user $u \in N_U$, returns the predicted label $p^S(u)$ and the confidence value $c^S(u)$, according to the spatial dimension.

3.4. Model fusion

The goal of the final stage is to combine the output of the different models learned from the textual content, from the network of relationships and from the spatial dimension, to get the final classification for each unlabeled user $u \in N_U$.

In SAIRUS, we perform this step by learning a model for combining the output of such models. This is done by resorting to a *Multi-Layer Perceptron* (MLP) used in a Stacked Generalization fashion [22]. MLP is a feedforward *Artificial Neural Network* (ANN) composed by an input layer, multiple hidden layers and an output layer, where the training occurs by iteratively updating the weights of the network through backpropagation [85].

The input layer of the adopted MLP consists of 7 neurons, that take as inputs, for a given user u: *i)* the reconstruction error of the safe autoencoder $AS(u)$ and of the risky autoencoder $AR(u)$, as well as the predicted label $p^c(u)$, obtained by the component for the semantic analysis of the textual content; *ii*) the predicted label $p^R(u)$ and the confidence value $c^R(u)$ obtained from the component for the analysis of the network of relationships; *iii)* the predicted label $p^S(u)$ and the confidence value $c^S(u)$ obtained from the component for the spatial analysis.

Formally, the final label $l(u)$ for a given unlabeled user u is computed as:

$$l(u) = MLP(AS(u), AR(u), p^c(u), p^R(u), c^R(u), p^S(u), c^S(u)) \qquad (7)$$

The adopted MLP architecture is shown in the bottom part of Figure 1. In the hidden layer we adopt the *sigmoid* activation function, since it allows to capture possible nonlinear dependencies occurring between input and output variables. On the other hand, the output layer exploits the *softmax* activation function. This choice is motivated by its well-known ability of dealing with classification tasks, since it predicts a multinomial probability distribution which is then leveraged to select the final class, according to the highest probability. Coherently, the class attribute for training examples is subject to one-hot-encoding [86], so that $\langle 1, 0\rangle$ in the output neurons represents that the user is safe, while $\langle 0, 1\rangle$ in the output neurons represents that the user is risky.

Coherently, the implemented MLP exploits the log loss function, that has shown to be effective for binary classification tasks [87]. Specifically, the log loss function measures how much the prediction probability is close to the corresponding true value.

We stress the fact that our approach, based on the stacked generalization framework, learns how to combine the outputs of three different models, without any user-defined criteria. Moreover, since it is not based on ensemble techniques, that would solely rely on the predictions $p^C(u)$, $p^R(u)$ and $p^S(u)$, SAIRUS is able to consider additional features, such as the reconstruction errors $AS(u)$ and $AR(u)$ and the prediction confidence $c^R(u)$ and $c^S(u)$.

4. Experiments

In the following subsections, we first describe the dataset considered in the evaluation of the performance achieved by SAIRUS. Then, we outline the experimental setting and describe the considered competitors. Finally, we show and discuss the obtained results.

4.1. Datasets

For the evaluation of the SAIRUS performances, we adopted a real-world Twitter dataset[2], retrieved through a compliant crawling system, and by relying on the Conditional Independence Coupling (CIC) algorithm to obtain a representative sample of users, with no specific hashtag, from the United States. Each tweet is associated with a sentiment score, i.e., an integer value which represents its polarity, computed through Stanford CoreNLP Toolkit [88], and manually revised by 3 domain experts.

The ground truth for the user label (i.e., *risky* or *safe*) has been built following two different strategies:

- **Keywords**. We mark a tweet as *risky* if it contains at least one of the keywords appearing in two manually curated lists, related to terrorism and threats[3], and to hate against immigrants and women[4]. We compute a score for each user as the ratio between the number of tweets marked as risky and the total number of tweets, assuming that users who post the majority of tweets containing words related to terrorism, threats and hate, are more likely to be *risky*.
- **Sentiment.** We assign a score to each user, computed as the sum of the sentiment score of their tweets. In this case, the main assumption is that users who post multiple tweets with a negative sentiment are more likely to be *risky*.

In both cases, we sort users according to their score and let three expert reviewers perform a manual inspection of their tweets, focusing on the top and on the bottom of the sorted list. Accordingly, a selection of the *safest* and of the *riskiest* users was performed. This process ensures the correctness of the labeling procedure, avoiding incorrect labels in the ground truth (more likely occurring for users in the middle of the list) that would have possibly led to misleading conclusions in the performance evaluation.

We performed an additional operation to inject noisy data under controlled conditions. Specifically, we injected *borderline* users who, in this case, may correspond to journalists who share negative textual contents for informative purposes, but are mainly connected with *safe* users. Specifically, risky users showing the majority of their neighbors in the network labeled as *safe* were considered as *borderline* and relabeled as *safe*. Finally, we removed users not connected with any other users. The quantitative characteristics of the obtained datasets are summarized in Table 1.

---

[2] According to the Twitter policies, the dataset cannot be publicly shared, but can be provided for research and reproducibility purposes upon request.

[3] `https://www.dailymail.co.uk/news/article-2150281/`

[4] `https://github.com/msang/hateval`

Table 1: Quantitative characteristics of the datasets based on *Keywords* and on *Sentiment*

	Keywords	**Sentiment**
Safe Users	1467	1470
Risky Users	2241	1033
Borderline Users	263	304
Tweets	7,686,231	10,016,749

4.2. Experimental setting and competitors

We evaluated the results obtained by SAIRUS with different dimensionality reduction techniques. Specifically, we adopted *PCA* [78], *Node2Vec* [81] and *Autoencoders bottleneck encodings* [80]. We also evaluated the results obtained with different values for the embedding dimensionality, namely k_c, for the semantic analysis of the textual content, k_r, for the analysis of the network of relationships, and k_s for the spatial analysis. Specifically, after performing some preliminary evaluations, we selected the following combinations of such parameters to perform the complete experiments: $\langle k_c$=128, k_r=256, k_s=256$\rangle$, $\langle k_c$=256, k_r=128, k_s=128$\rangle$, and $\langle k_c$=512, k_r=128, k_s=128$\rangle$.

The results obtained by SAIRUS were compared with those achieved by several competitors. Specifically, we evaluated the performance achieved by a classifier based on Random Forests (**RF**) with 100 trees, by optimizing the minimal cost-complexity pruning parameter α in $\{0.0, 0.2, 0.5, 1.0, 2.0\}$. Moreover, for the content-based analysis (coherently with the approach followed by SAIRUS), we also adopted two one-class classifiers based on autoencoders (**1C-AEs**).

The models were trained starting from different sets of features, each exploiting one, namely content (C), relationships (R) or spatial (S), or more (C+R, C+S, R+S, and C+R+S) perspectives. When more than one perspective was considered, we built the feature set as the concatenation of the feature sets associated with each single perspective. As state-of-the-art systems to build the feature set from the textual content, we considered *Word2Vec* (**w2v**) [39] and *Doc2Vec* (**d2v**) [15]. In this case, coherently with the setting adopted for SAIRUS, we set their embedding dimensionality to the same value adopted for k_c. On the other hand, in order to learn a feature representation from the network of relationships and from the spatial closeness network, we adopted the system *Node2Vec* (**n2v**) [81]. Also in this case, coherently with the setting adopted for SAIRUS, the embedding dimensionality was set to k_r and k_s, respectively. Overall, we compared SAIRUS with 13 competitors (see Table 2).

All the experiments were carried out on a server equipped with a Xeon CPU E5-1650-v3 and 64 GB of RAM. We adopted a stratified 5-fold cross-validation, randomly partitioning the users into 5 folds and alternatively selecting one fold as testing set (N_U) and the remaining 4 folds as training set (N_L). The adopted stratification allowed us to preserve the ratio of safe and risky users, as well as the ratio of bordeline users within safe users. As evaluation measures, we used *precision*, *recall*, *F1-Score*, and *accuracy*, considering the risky label as positive class. We also evaluated such measures on the borderline users, with the purpose of assessing the effectiveness of the methods when dealing with noisy data.

Table 2: Summary of the considered competitors.

Classifier	C	R	S
1C-AEs	✓(d2v)		
1C-AEs	✓(w2v)		
RF	✓(d2v)		
RF	✓(w2v)		
RF		✓	
RF			✓
RF	✓(d2v)	✓	
RF	✓(w2v)	✓	
RF	✓(d2v)		✓
RF	✓(w2v)		✓
RF		✓	✓
RF	✓(d2v)	✓	✓
RF	✓(w2v)	✓	✓

4.3. Results and discussion

In Tables 3-5 and 6-8, we show the results obtained on the *sentiment* dataset and on the *keywords* dataset, respectively, where we emphasize (in bold, with gray background) the best result obtained for a given evaluation measure (column of the table). We start our discussion by looking at the results obtained by the competitors. Focusing on the solutions solely based on the textual content, we can observe that the adoption of w2v generally leads to better results with respect to d2v. Although d2v is able to directly represent whole documents by introducing a unique document *id* instead of aggregating the word embeddings [15], the superiority of w2v has been already shown in several contexts (see, for example, [89]), mainly due to its ability of modeling different topics spread over different paragraphs, that generally reduces overfitting issues. As regards the classifiers, we can see that RF and 1C-AEs lead to comparable results, with no solution clearly dominating the other. The adoption of features related to user relationships (R) or to the spatial dimension (S) does not seem to provide a clear contribution to competitors. Indeed, none of the more complex feature sets led to higher values for F1-score or accuracy than the one solely based on the textual content. This result confirms that simply injecting features coming from one perspective into the other could also compromise the results due to the possible introduction of issues related to the course of dimensionality. The situation slightly changes when looking at the borderline users. Indeed, in this case, the contribution coming from the features based on user relationships support the competitors in making more informed predictions about this kind of users. This situation appears coherent along the different values adopted for k_c, k_r and k_s, as well as over the two different considered datasets.

On the other hand, looking at the performance exhibited by SAIRUS, we can immediately notice that the best results are obtained when the network of user relationships or the spatial dimension (or both) is exploited. This aspect is more evident on the dataset *sentiment*, where the achieved F1-score, when both user relationships and the spatial analysis are considered, is ~ 0.8. This confirms that the approach adopted by SAIRUS to fuse the contribution coming from multiple perspectives is much more effective than the concatenation of features.

In the dataset based on keywords, we can observe a more balanced situation, where the configuration that exploits the textual content and the spatial analysis C+S slightly emerges as the best one, with comparable results obtained by the C+R configuration. These results confirm the relevance of the spatial perspective, as well as the importance of properly modeling and exploiting it through a smart fusion strategy. Similar conclusions can be drawn for the borderline users (i.e., journalists). Indeed, independently from the embedding dimensions and the chosen network representation, the best results are obtained when the spatial dimension is considered.

Focusing on the embedding parameters (k_c, k_r and k_s), it appears that adopting a wider feature vector for the textual content (k_c) provides benefits in terms of F1-score. This is also confirmed by the overall best results achieved in the setting $\langle k_c$=512, k_r=128, k_s=128$\rangle$. As regards the dimensionality reduction, PCA and Autoencoders led to the best results with an average F1-score of ~ 0.7.

A deeper analysis of the influence of the considered perspectives, of the embedding parameters, and of the strategy adopted to reduce the dimensionality of the adjacency matrices can be done by observing Figure 3. From this figure, we can easily conclude that considering both the network of user relationships and the spatial dimension generally leads to the best results. Moreover, as already mentioned, the highest value for k_c, namely $k_c = 512$, led to the best results, while the best value for k_r and k_s appears to be the lowest among the considered ones (i.e., 128). These results can be motivated by the richness and heterogeneity of the topics of the tweets, that need a higher dimensionality of the feature space to be properly represented. On the other hand, the network of relationships and of spatial closeness are quite sparse, and a low-dimension feature space appears to be adequate. As for the strategy adopted to reduce the dimensionality, the autoencoder appears to be the clear winner, with general better results and a significantly lower variance.

The results achieved by SAIRUS, when compared to those obtained by competitors, are much higher, according to all the evaluation measures, on both the considered datasets. This is true both when analyzing the whole set of users and when focusing on borderline users, and emphasizes the capability of SAIRUS of being robust to noisy users, while keeping a generally high predictive accuracy. This is clearly due to the hybrid approach we adopt where every perspective can be used to provide confirmations on what predicted by other perspectives. Moreover, the ability of fruitfully capturing the information conveyed by the network of relationships and by the spatial closeness among users makes SAIRUS a state-of-the-art tool to properly distinguish between risky and safe users in a social network, and envisages its adoption to properly exploit the massive amount of data currently generated from geo-located mobile devices.

Table 3: Results on the *sentiment* dataset, with $k_c = 128, k_r = 256, k_s = 256$

		Configuration				All users				Borderline			
		Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS		1C-AEs	✓(d2v)			0.557	0.565	0.540	0.563	0.500	0.233	0.317	0.467
		1C-AEs	✓(w2v)			0.650	0.646	0.666	0.648	0.500	0.132	0.207	0.263
		RF	✓(d2v)			0.500	0.450	0.473	0.600	0.575	0.526	0.492	0.677
		RF	✓(w2v)			0.687	0.686	0.686	0.686	0.500	0.179	0.263	0.358
		RF		✓		0.503	0.501	0.473	0.642	0.500	0.455	0.476	0.910
		RF			✓	0.478	0.494	0.450	0.647	0.500	0.463	0.481	0.927
		RF	✓(d2v)	✓		0.568	0.509	0.441	0.681	0.600	0.583	0.591	0.967
		RF	✓(w2v)	✓		0.681	0.680	0.680	0.680	0.500	0.146	0.225	0.292
		RF	✓(d2v)		✓	0.515	0.508	0.491	0.635	0.500	0.423	0.458	0.847
		RF	✓(w2v)		✓	0.602	0.602	0.602	0.602	0.500	0.198	0.283	0.396
		RF		✓	✓	0.502	0.501	0.480	0.632	0.500	0.437	0.466	0.873
		RF	✓(d2v)	✓	✓	0.514	0.506	0.480	0.645	0.500	0.422	0.456	0.843
		RF	✓(w2v)	✓	✓	0.607	0.607	0.607	0.607	0.500	0.179	0.263	0.358
SAIRUS	**Dimensionality Reduction**	AE	✓	✓		0.591	0.723	0.643	0.723	1.000	0.943	0.970	0.943
			✓		✓	0.657	0.748	0.690	0.748	1.000	0.953	0.976	0.953
				✓	✓	0.773	0.781	0.772	0.781	1.000	0.820	0.900	0.820
			✓	✓	✓	0.720	0.766	0.727	0.766	1.000	0.977	0.988	0.977
		Node2vec	✓	✓		0.671	0.756	0.704	0.756	1.000	0.847	0.912	0.847
			✓		✓	0.603	0.718	0.643	0.718	1.000	0.967	0.983	0.967
				✓	✓	0.761	0.757	0.758	0.757	1.000	0.690	0.816	0.690
			✓	✓	✓	0.611	0.741	0.660	0.741	1.000	0.960	0.978	0.960
		PCA	✓	✓		0.671	0.759	0.703	0.759	1.000	0.900	0.945	0.900
			✓		✓	0.514	0.648	0.568	0.648	1.000	0.973	0.986	0.973
				✓	✓	0.785	**0.791**	**0.784**	**0.791**	1.000	0.857	0.922	0.857
			✓	✓	✓	0.743	0.740	0.695	0.740	**1.000**	**0.980**	**0.990**	**0.980**
		None	✓	✓		0.735	0.749	0.686	0.749	1.000	0.970	0.984	0.970
			✓		✓	0.576	0.625	0.596	0.625	1.000	0.937	0.967	0.937
				✓	✓	**0.793**	0.768	0.727	0.768	1.000	0.950	0.974	0.950
			✓	✓	✓	0.711	0.707	0.664	0.707	1.000	0.907	0.946	0.907

Table 4: Results on the *sentiment* dataset, with $k_c = 256, k_r = 128, k_s = 128$

		Configuration				All users				Borderline			
		Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS		1C-AEs	✓(d2v)			0.564	0.568	0.557	0.595	0.500	0.277	0.351	0.553
		1C-AEs	✓(w2v)			0.699	0.671	0.682	0.720	0.500	0.202	0.285	0.403
		RF	✓(d2v)			0.577	0.530	0.502	0.676	0.500	0.437	0.466	0.873
		RF	✓(w2v)			0.687	0.686	0.686	0.686	0.500	0.165	0.248	0.331
		RF		✓		0.508	0.504	0.474	0.646	0.500	0.445	0.471	0.890
		RF			✓	0.498	0.499	0.464	0.645	0.500	0.458	0.478	0.917
		RF	✓(d2v)	✓		0.500	0.462	0.480	0.623	0.580	0.518	0.466	0.681
		RF	✓(w2v)	✓		0.681	0.680	0.680	0.680	0.500	0.146	0.225	0.292
		RF	✓(d2v)		✓	0.540	0.523	0.509	0.649	0.500	0.430	0.462	0.860
		RF	✓(w2v)		✓	0.602	0.602	0.602	0.602	0.500	0.198	0.283	0.396
		RF		✓	✓	0.503	0.502	0.479	0.636	0.500	0.430	0.462	0.860
		RF	✓(d2v)	✓	✓	0.530	0.515	0.491	0.652	0.500	0.432	0.463	0.863
		RF	✓(w2v)	✓	✓	0.607	0.607	0.607	0.607	0.500	0.179	0.263	0.358
SAIRUS	**Dimensionality Reduction**	AE	✓	✓		0.720	0.747	0.692	0.747	1.000	0.910	0.951	0.910
			✓		✓	0.688	0.767	0.713	0.767	1.000	0.800	0.814	0.800
				✓	✓	0.776	0.783	0.776	0.783	1.000	0.853	0.920	0.853
			✓	✓	✓	0.667	0.755	0.695	0.755	**1.000**	**0.980**	**0.990**	**0.980**
		Node2vec	✓	✓		0.644	0.710	0.634	0.710	1.000	0.897	0.940	0.897
			✓		✓	0.591	0.719	0.643	0.719	1.000	0.973	0.986	0.973
				✓	✓	0.754	0.755	0.754	0.755	1.000	0.763	0.865	0.763
			✓	✓	✓	0.759	**0.801**	0.767	**0.801**	1.000	0.943	0.969	0.943
		PCA	✓	✓		0.725	0.752	0.697	0.752	1.000	0.917	0.955	0.917
			✓		✓	0.593	0.694	0.619	0.694	1.000	0.773	0.803	0.773
				✓	✓	0.786	0.793	**0.785**	0.793	1.000	0.860	0.924	0.860
			✓	✓	✓	0.711	0.702	0.682	0.702	1.000	0.923	0.958	0.923
		None	✓	✓		0.742	0.770	0.720	0.770	1.000	0.837	0.884	0.837
			✓		✓	0.550	0.639	0.586	0.639	1.000	0.960	0.979	0.960
				✓	✓	**0.784**	0.768	0.727	0.768	1.000	0.950	0.974	0.950
			✓	✓	✓	0.711	0.702	0.682	0.702	1.000	0.923	0.958	0.923

Table 5: Results on the *sentiment* dataset, with $k_c = 512, k_r = 128, k_s = 128$

		Configuration				All users				Bordeline			
		Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS		1C-AEs	✓(d2v)			0.567	0.546	0.540	0.654	0.500	0.363	0.419	0.727
		1C-AEs	✓(w2v)			0.612	0.604	0.605	0.635	0.500	0.155	0.233	0.310
		RF	✓(d2v)			0.562	0.524	0.490	0.672	0.500	0.443	0.469	0.887
		RF	✓(w2v)			0.687	0.686	0.686	0.686	0.500	0.165	0.248	0.331
		RF		✓		0.508	0.504	0.474	0.646	0.500	0.445	0.471	0.890
		RF			✓	0.498	0.499	0.464	0.645	0.500	0.458	0.478	0.917
		RF	✓(d2v)	✓		0.559	0.517	0.471	0.676	0.500	0.455	0.476	0.910
		RF	✓(w2v)	✓		0.681	0.680	0.680	0.680	0.500	0.146	0.225	0.292
		RF	✓(d2v)		✓	0.535	0.521	0.505	0.648	0.500	0.410	0.450	0.820
		RF	✓(w2v)		✓	0.602	0.602	0.602	0.602	0.500	0.198	0.283	0.396
		RF		✓	✓	0.503	0.502	0.479	0.636	0.500	0.430	0.462	0.860
		RF	✓(d2v)	✓	✓	0.539	0.519	0.498	0.653	0.500	0.428	0.461	0.857
		RF	✓(w2v)	✓	✓	0.607	0.607	0.607	0.607	0.500	0.179	0.263	0.358
SAIRUS	**Dimensionality Reduction**	AE	✓	✓		0.777	0.784	0.776	0.784	1.000	0.853	0.920	0.853
			✓		✓	**0.814**	**0.816**	**0.810**	**0.816**	1.000	0.847	0.890	0.847
				✓	✓	0.776	0.783	0.776	0.783	1.000	0.853	0.920	0.853
			✓	✓	✓	0.806	0.807	0.795	0.807	1.000	0.940	0.968	0.940
		Node2vec	✓	✓		0.757	0.758	0.757	0.758	1.000	0.770	0.868	0.770
			✓		✓	0.710	0.754	0.728	0.754	**1.000**	**0.970**	**0.985**	**0.970**
				✓	✓	0.776	0.772	0.774	0.772	1.000	0.787	0.879	0.787
			✓	✓	✓	0.788	0.786	0.773	0.786	1.000	0.953	0.976	0.953
		PCA	✓	✓		0.790	0.797	0.789	0.797	1.000	0.860	0.924	0.860
			✓		✓	0.566	0.612	0.585	0.612	1.000	0.943	0.970	0.943
				✓	✓	0.786	0.793	0.785	0.793	1.000	0.860	0.924	0.860
			✓	✓	✓	0.751	0.741	0.691	0.741	1.000	0.967	0.983	0.967
		None	✓	✓		0.800	0.779	0.744	0.779	1.000	0.877	0.925	0.877
			✓		✓	0.636	0.639	0.637	0.639	1.000	0.850	0.911	0.850
				✓	✓	0.793	0.768	0.727	0.768	1.000	0.950	0.974	0.950
			✓	✓	✓	0.755	0.719	0.655	0.719	1.000	0.967	0.983	0.967

Table 6: Results on the *keywords* dataset, with $k_c = 128, k_r = 256, k_s = 256$

		Configuration				All users				Bordeline			
		Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS		1C-AEs	✓(d2v)			0.547	0.546	0.544	0.546	0.500	0.219	0.298	0.438
		1C-AEs	✓(w2v)			0.637	0.631	0.630	0.631	0.500	0.238	0.318	0.477
		RF	✓(d2v)			0.559	0.559	0.559	0.559	0.500	0.215	0.297	0.431
		RF	✓(w2v)			0.687	0.686	0.686	0.686	0.500	0.165	0.248	0.331
		RF		✓		0.496	0.496	0.494	0.496	0.500	0.254	0.337	0.508
		RF			✓	0.511	0.511	0.509	0.511	0.500	0.225	0.309	0.450
		RF	✓(d2v)	✓		0.567	0.567	0.566	0.567	0.500	0.221	0.303	0.442
		RF	✓(w2v)	✓		0.681	0.680	0.680	0.680	0.500	0.146	0.225	0.292
		RF	✓(d2v)		✓	0.544	0.544	0.543	0.544	0.500	0.231	0.313	0.462
		RF	✓(w2v)		✓	0.602	0.602	0.602	0.602	0.500	0.198	0.283	0.396
		RF		✓	✓	0.489	0.490	0.488	0.489	0.500	0.256	0.338	0.512
		RF	✓(d2v)	✓	✓	0.543	0.543	0.541	0.543	0.500	0.235	0.315	0.469
		RF	✓(w2v)	✓	✓	0.623	0.623	0.623	0.623	0.500	0.173	0.233	0.347
SAIRUS	**Dimensionality Reduction**	AE	✓	✓		0.599	0.862	0.696	0.616	0.600	0.419	0.493	0.419
			✓		✓	0.620	**0.870**	0.711	0.636	0.600	0.569	0.584	0.569
				✓	✓	0.667	0.766	0.713	0.691	1.000	0.700	0.822	0.700
			✓	✓	✓	0.632	0.858	0.711	0.641	0.600	0.538	0.565	0.538
		Node2vec	✓	✓		0.608	0.794	0.668	0.605	0.600	0.404	0.482	0.404
			✓		✓	0.657	0.824	0.708	0.648	0.600	0.492	0.538	0.492
				✓	✓	0.689	0.676	0.682	0.685	1.000	0.596	0.746	0.596
			✓	✓	✓	0.717	0.684	0.677	0.672	0.800	0.669	0.721	0.669
		PCA	✓	✓		0.618	0.867	0.709	0.633	0.600	0.462	0.522	0.462
			✓		✓	0.528	0.693	0.577	0.525	0.600	0.546	0.572	0.546
				✓	✓	0.687	0.776	**0.729**	**0.711**	1.000	0.746	0.854	0.746
			✓	✓	✓	0.682	0.661	0.645	0.646	0.800	0.727	0.760	0.727
		None	✓	✓		0.749	0.568	0.526	0.578	0.600	0.565	0.582	0.565
			✓		✓	0.543	0.696	0.585	0.537	0.600	0.527	0.561	0.527
				✓	✓	**0.892**	0.317	0.468	0.665	**1.000**	**0.938**	**0.968**	**0.953**
			✓	✓	✓	0.666	0.424	0.443	0.562	0.800	0.781	0.790	0.781

Table 7: Results on the *keywords* dataset, with $k_c = 256, k_r = 128, k_s = 128$

	Configuration				**All users**				**Bordeline**			
	Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS	1C-AEs	✓(d2v)			0.552	0.551	0.547	0.551	0.500	0.204	0.284	0.408
	1C-AEs	✓(w2v)			0.640	0.637	0.637	0.637	0.500	0.229	0.311	0.458
	RF	✓(d2v)			0.568	0.568	0.568	0.568	0.500	0.237	0.321	0.473
	RF	✓(w2v)			0.688	0.687	0.687	0.687	0.500	0.158	0.239	0.315
	RF		✓		0.478	0.478	0.476	0.478	0.500	0.273	0.353	0.546
	RF			✓	0.502	0.502	0.501	0.502	0.500	0.277	0.354	0.554
	RF	✓(d2v)	✓		0.565	0.565	0.564	0.565	0.500	0.202	0.285	0.404
	RF	✓(w2v)	✓		0.690	0.688	0.688	0.689	0.500	0.152	0.232	0.304
	RF	✓(d2v)		✓	0.561	0.560	0.558	0.560	0.500	0.244	0.326	0.488
	RF	✓(w2v)		✓	0.628	0.628	0.628	0.628	0.500	0.183	0.267	0.365
	RF		✓	✓	0.487	0.487	0.487	0.487	0.500	0.254	0.336	0.508
	RF	✓(d2v)	✓	✓	0.571	0.571	0.570	0.571	0.500	0.246	0.328	0.492
	RF	✓(w2v)	✓	✓	0.646	0.646	0.646	0.646	0.500	0.175	0.258	0.350
SAIRUS — **Dimensionality Reduction**	AE	✓	✓		0.637	**0.812**	0.706	0.657	0.800	0.558	0.656	0.558
		✓		✓	0.657	0.808	0.715	0.672	0.800	0.758	0.778	0.758
			✓	✓	0.673	0.774	0.720	0.698	1.000	0.700	0.822	0.700
		✓	✓	✓	0.730	0.732	0.731	**0.730**	1.000	0.919	0.957	0.919
	Node2vec	✓	✓		0.649	0.761	0.689	0.654	0.800	0.454	0.578	0.454
		✓		✓	0.681	0.747	0.698	0.672	0.800	0.750	0.774	0.750
			✓	✓	0.701	0.672	0.686	0.692	1.000	0.677	0.807	0.677
		✓	✓	✓	0.579	0.505	0.533	0.649	1.000	0.835	0.895	0.835
	PCA	✓	✓		0.652	0.811	0.713	0.668	0.800	0.635	0.708	0.635
		✓		✓	0.595	0.684	0.623	0.598	0.800	0.619	0.686	0.619
			✓	✓	0.692	0.777	**0.731**	0.714	1.000	0.785	0.879	0.785
		✓	✓	✓	0.553	0.534	0.525	0.635	1.000	0.869	0.926	0.869
	None	✓	✓		0.822	0.546	0.572	0.652	0.800	0.619	0.670	0.619
		✓		✓	0.539	0.711	0.593	0.539	0.600	0.531	0.563	0.531
			✓	✓	**0.916**	0.295	0.445	0.633	**1.000**	**0.942**	**0.970**	**0.942**
		✓	✓	✓	0.859	0.446	0.504	0.625	1.000	0.796	0.804	0.796

Table 8: Results on the *keywords* dataset, with $k_c = 512, k_r = 128, k_s = 128$

	Configuration				**All users**				**Bordeline**			
	Classifier	**C**	**R**	**S**	**Prec**	**Rec**	**F1**	**Acc**	**Prec**	**Rec**	**F1**	**Acc**
COMPETITORS	1C-AEs	✓(d2v)			0.550	0.549	0.546	0.549	0.500	0.208	0.289	0.415
	1C-AEs	✓(w2v)			0.635	0.630	0.629	0.630	0.500	0.231	0.311	0.462
	RF	✓(d2v)			0.570	0.570	0.569	0.570	0.500	0.238	0.320	0.477
	RF	✓(w2v)			0.689	0.688	0.688	0.688	0.500	0.158	0.238	0.315
	RF		✓		0.478	0.478	0.476	0.478	0.500	0.273	0.353	0.546
	RF			✓	0.502	0.502	0.501	0.502	0.500	0.277	0.354	0.554
	RF	✓(d2v)	✓		0.576	0.576	0.576	0.576	0.500	0.237	0.317	0.473
	RF	✓(w2v)	✓		0.689	0.687	0.687	0.687	0.500	0.146	0.225	0.292
	RF	✓(d2v)		✓	0.556	0.556	0.555	0.556	0.500	0.246	0.326	0.492
	RF	✓(w2v)		✓	0.650	0.650	0.650	0.650	0.500	0.177	0.260	0.354
	RF		✓	✓	0.487	0.487	0.487	0.487	0.500	0.254	0.336	0.508
	RF	✓(d2v)	✓	✓	0.557	0.557	0.556	0.557	0.500	0.223	0.302	0.446
	RF	✓(w2v)	✓	✓	0.654	0.653	0.653	0.653	0.500	0.185	0.268	0.369
SAIRUS — **Dimensionality Reduction**	AE	✓	✓		0.674	0.776	0.721	0.699	1.000	0.700	0.822	0.700
		✓		✓	0.705	0.779	0.740	0.726	1.000	0.950	0.974	0.950
			✓	✓	0.673	0.774	0.720	0.698	1.000	0.700	0.822	0.700
		✓	✓	✓	0.688	0.767	0.712	0.685	1.000	0.777	0.794	0.777
	Node2vec	✓	✓		0.686	0.686	0.686	0.685	1.000	0.558	0.715	0.558
		✓		✓	0.725	0.680	0.702	0.710	1.000	0.931	0.964	0.931
			✓	✓	0.705	0.697	0.701	0.701	1.000	0.665	0.799	0.665
		✓	✓	✓	0.780	0.606	0.674	0.710	1.000	0.915	0.954	0.915
	PCA	✓	✓		0.732	**0.803**	**0.765**	**0.752**	1.000	0.662	0.780	0.662
		✓		✓	0.555	0.506	0.529	0.549	1.000	0.900	0.947	0.900
			✓	✓	0.692	0.777	0.731	0.714	1.000	0.785	0.879	0.785
		✓	✓	✓	0.754	0.600	0.659	0.697	1.000	0.904	0.947	0.904
	None	✓	✓		0.914	0.297	0.448	0.634	1.000	0.938	0.968	0.938
		✓		✓	0.579	0.518	0.546	0.569	1.000	0.908	0.951	0.908
			✓	✓	**0.916**	0.295	0.445	0.633	1.000	0.942	0.970	0.942
		✓	✓	✓	0.878	0.305	0.431	0.622	**1.000**	**0.988**	**0.994**	**0.988**

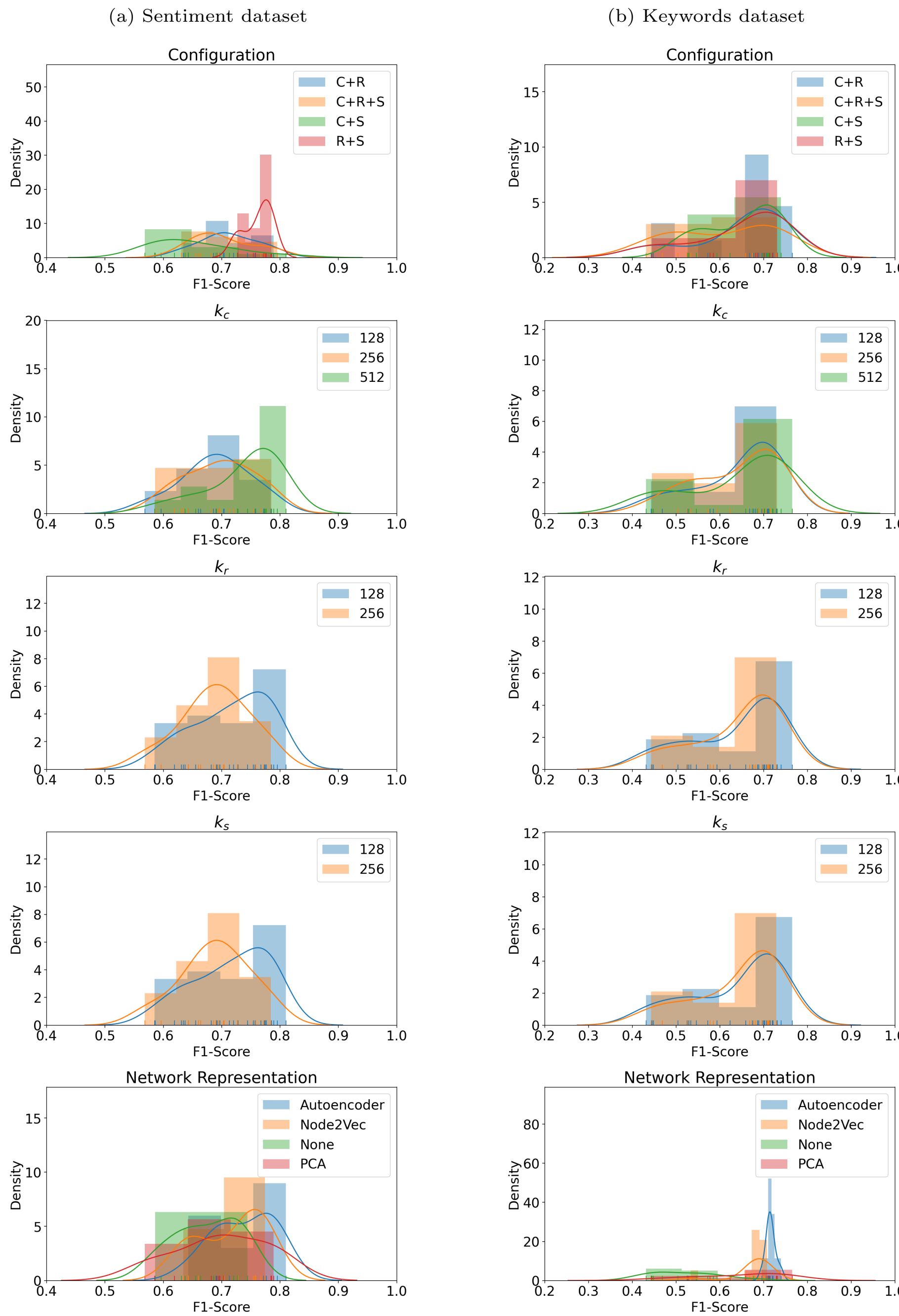

Figure 3: Analysis of the influence of the considered perspectives, of the embedding parameters, and of the strategy adopted to reduce the dimensionality of the adjacency matrices.

5. Conclusion

In this paper, we proposed a novel method for the identification of *risky* users in social networks, called SAIRUS. The proposed method falls in the category of hybrid approaches for node classification in network data, since it is able to fruitfully exploit and fuse the contribution of different perspectives of social network data. Specifically SAIRUS takes into account *i)* the semantics conveyed by the textual content posted by the users, *ii)* the network of user relationships, and *iii)* the spatial closeness among users. To the best of the authors knowledge, this is the first approach that simultaneously takes into account all these dimensions of analysis. Moreover, contrary to existing methods, SAIRUS specifically exploits the peculiarities of each kind of data, without falling back into feature injection approaches.

The performance obtained by SAIRUS was evaluated on two versions of a real-world Twitter dataset, and compared against 13 competitors that consider either one perspective at a time or a combination thereof. In all the situations, the results exhibited by SAIRUS demonstrated to be superior to all the considered competitors, and very robust to the presence of noisy users, in terms of all the evaluation measures.

Note that SAIRUS is also able to implicitly take into account the temporal dimension related to the textual content, but currently cannot consider the dynamism of the network of relationships or the dynamism of the spatial closeness among users. For future work, we will focus on making SAIRUS able to specifically capture these aspects, allowing it to detect users with a safe background or history, who suddenly start to post negative contents, or join risky communities. We also plan to extend the framework to support the analysis of other types of unstructured content, such as images or videos. Finally, we will consider the design of a distributed version of SAIRUS implemented in Apache Spark, in order to make it able to analyze large scale networks.

6. Acknowledgments

The authors acknowledge the support of the European Commission through the H2020 Project "CounteR - Privacy-First Situational Awareness Platform for Violent Terrorism and Crime Prediction, Counter Radicalisation and Citizen Protection" (Grant N. 101021607).

References

[1] S. Tabassum, F. S. Pereira, S. Fernandes, J. Gama, Social network analysis: An overview, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8 (5) (2018).

[2] K. Zhang, S. Bhattacharyya, S. Ram, Large-scale network analysis for online social brand advertising., Mis Quarterly 40 (4) (2016).

[3] J. Vithayathil, M. Dadgar, J. K. Osiri, Social media use and consumer shopping preferences, International Journal of Information Management 54 (2020) 102117.

[4] T. Radicioni, F. Saracco, E. Pavan, T. Squartini, Analysing twitter semantic networks: the case of 2018 italian elections, Scientific Reports 11 (1) (2021) 1–22.

[5] M. Chary, N. Genes, A. McKenzie, A. F. Manini, Leveraging social networks for toxicovigilance, Journal of Medical Toxicology 9 (2) (2013) 184–191.

[6] E. Ferrara, Contagion dynamics of extremist propaganda in social networks, Information Sciences 418 (2017) 1–12.

[7] B. Huang, E. Raisi, Online Harassment, Springer International Publishing, Cham, 2018, Ch. Weak Supervision and Machine Learning for Online Harassment Detection, pp. 5–28.

[8] I. Awan, Cyber-Extremism: Isis and the Power of Social Media, Society 54 (2) (2017) 138–149.

[9] A. Al-Rawi, J. Groshek, Jihadist Propaganda on Social Media: An Examination of ISIS Related Content on Twitter, International Journal of Cyber Warfare and Terrorism (IJCWT) 8 (4) (2018) 1–15.

[10] M. Alfifi, P. Kaghazgaran, J. Caverlee, F. Morstatter, A Large-Scale Study of ISIS Social Media Strategy: Community Size, Collective Influence, and Behavioral Impact, Proc. of the International AAAI Conference on Web and Social Media 13 (2019) 58–67.

[11] J. Thee, I. Alsmadi, S. Al-khateeb, Pro-isis tweets analysis using machine learning techniques, in: 2020 IEEE International Conference on Big Data (Big Data), 2020, pp. 4351–4358.

[12] W. Zhou, C. Han, X. Huang, Multiclass classification of tweets and twitter users based on kindness analysis, in: CS229 Final Project Report, 2016.

[13] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: Proc. of the Conference of the NAACL 2016: Human Language Technologies, Association for Computational Linguistics, 2016, pp. 1480–1489.

[14] V. N. Uzel, E. Saraç Eşsiz, S. Ayşe Özel, Using fuzzy sets for detecting cyber terrorism and extremism in the text, in: 2018 Innovations in Intelligent Systems and Applications Conference (ASYU), 2018, pp. 1–4.

[15] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, 2014, pp. 1188–1196.

[16] M. Ji, Y. Sun, M. Danilevsky, J. Han, J. Gao, Graph regularized transductive classification on heterogeneous information networks, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 570–586.

[17] S. A. Macskassy, F. Provost, Classification in networked data: A toolkit and a univariate case study, Journal of machine learning research 8 (May) (2007) 935–983.

[18] B. Gallagher, H. Tong, T. Eliassi-Rad, C. Faloutsos, Using ghost edges for classification in sparsely labeled networks, in: Proc. of SIGKDD int. conference on Knowledge discovery and data mining, ACM, 2008, pp. 256–264.

[19] M. Bilgic, L. Getoor, Effective label acquisition for collective classification, in: Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '08, ACM, 2008, pp. 43–51.

[20] M. Mateen, M. A. Iqbal, M. Aleem, M. A. Islam, A hybrid approach for spam detection for twitter, in: 2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 2017, pp. 466–471.

[21] T. Hamdi, H. Slimi, I. Bounhas, Y. Slimani, A hybrid approach for fake news detection in twitter based on user features and graph embedding, in: Distributed Computing and Internet Technology, Springer International Publishing, Cham, 2020, pp. 266–280.

[22] D. H. Wolpert, Stacked generalization, Neural Networks 5 (2) (1992) 241–259.

[23] G. Xu, J. Qi, D. Huang, M. Daneshmand, Detecting spammers on social networks based on a hybrid model, in: 2016 IEEE International Conference on Big Data (Big Data), 2016, pp. 3062–3068.

[24] B. Fields, K. Jacobson, C. Rhodes, M. d'Inverno, M. Sandler, M. Casey, Analysis and exploitation of musician social networks for recommendation and discovery, IEEE Transactions on Multimedia 13 (4) (2011) 674–686.

[25] D. Jin, X. Wang, R. He, D. He, J. Dang, W. Zhang, Robust detection of link communities in large social networks by exploiting link semantics, in: AAAI'18/IAAI'18/EAAI'18, AAAI Press, 2018, pp. 314—-321.

[26] J. Scott, Social network analysis, Sociology 22 (1) (1988) 109–127.

[27] S. P. Borgatti, B. Ofem, Social network theory and analysis, Social network theory and educational change (2010) 17–29.

[28] W. Jo, D. Chang, M. You, G.-H. Ghim, A social network analysis of the spread of covid-19 in south korea and policy implications, Scientific Reports 11 (1) (2021) 1–10.

[29] M. Windzio, The network of global migration 1990–2013: Using ergms to test theories of migration between countries, Social Networks 53 (2018) 20–29.

[30] V. Danchev, M. A. Porter, Neither global nor local: Heterogeneous connectivity in spatial network structures of world migration, Social Networks 53 (2018) 4–19.

[31] C. Intal, T. Yasseri, Dissent and rebellion in the house of commons: A social network analysis of brexit-related divisions in the 57th parliament, Applied Network Science 6 (1) (2021) 1–12.

[32] E. Wu, R. Carleton, G. Davies, Discovering bin-laden's replacement in al-qaeda, using social network analysis: A methodological investigation, Perspectives on Terrorism 8 (1) (2014) 57–73.

[33] P. Choudhary, U. Singh, A survey on social network analysis for counter-terrorism, International Journal of Computer Applications 112 (9) (2015) 24–29.

[34] I. Gialampoukidis, G. Kalpakis, T. Tsikrika, S. Vrochidis, I. Kompatsiaris, Key player identification in terrorism-related social media networks using centrality measures, in: 2016 European Intelligence and Security Informatics Conference (EISIC), 2016, pp. 112–115.

[35] G. Kalpakis, T. Tsikrika, S. Vrochidis, I. Kompatsiaris, Identifying terrorism-related key actors in multidimensional social networks, in: International Conference on Multimedia Modeling, Springer, 2019, pp. 93–105.

[36] G. Patil, K. Manwade, P. Landge, A novel approach for social network analysis & web mining for counter terrorism, International Journal on Computer Science and Engineering 4 (11) (2012) 1816.

[37] A. Sachan, Countering terrorism through dark web analysis, in: 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), 2012, pp. 1–5.

[38] S. M. Nagarajan, U. D. Gandhi, Classifying streaming of twitter data based on sentiment analysis using hybridization, Neural Computing and Applications 31 (5) (2019) 1425–1433.

[39] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).

[40] M. Bilgin, İ. F. Şentürk, Sentiment analysis on twitter data with semi-supervised doc2vec, in: 2017 international conference on computer science and engineering (UBMK), Ieee, 2017, pp. 661–666.

[41] L. Q. Trieu, H. Q. Tran, M.-T. Tran, News classification from social media using twitter-based doc2vec model and automatic query expansion, in: Proceedings of the Eighth International Symposium on Information and Communication Technology, SoICT 2017, Association for Computing Machinery, New York, NY, USA, 2017, p. 460–467.

[42] C. Van Hee, G. Jacobs, C. Emmery, B. Desmet, E. Lefever, B. Verhoeven, G. De Pauw, W. Daelemans, V. Hoste, Automatic detection of cyberbullying in social media text, PloS one 13 (10) (2018).

[43] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, the Journal of machine Learning research 3 (2003) 993–1022.

[44] L. Getoor, Link-based classification, in: Advanced methods for knowledge discovery from complex data, Springer, 2005, pp. 189–207.

[45] J. Neville, D. Jensen, Collective classification with relational dependency networks, in: Workshop on Multi-Relational Data Mining (MRDM-2003), 2003, p. 77.

[46] B. Taskar, P. Abbeel, D. Koller, Discriminative probabilistic models for relational data, arXiv preprint arXiv:1301.0604 (2012).

[47] M. Bilgic, L. Getoor, Active inference for collective classification, Proceedings of the AAAI Conference on Artificial Intelligence 24 (1) (2010) 1652–1655.

[48] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, T. Eliassi-Rad, Collective classification in network data, AI Magazine 29 (3) (2008) 93.

[49] W. Campbell, E. Baseman, K. Greenfield, Content+ context networks for user classification in twitter, in: Neural Information Processing Systems (NIPS) 2014 Workshop, 2013.

[50] M. A. Masood, R. A. Abbasi, Using graph embedding and machine learning to identify rebels on twitter, Journal of Informetrics 15 (1) (2021) 101121.

[51] G. Pio, F. Serafino, D. Malerba, M. Ceci, Multi-type clustering and classification from heterogeneous networks, Information Sciences 425 (2018) 107–126.

[52] M. Ceci, A. Appice, D. Malerba, Mr-sbc: A multi-relational naïve bayes classifier, in: N. Lavrač, D. Gamberger, L. Todorovski, H. Blockeel (Eds.), Knowledge Discovery in Databases: PKDD 2003, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 95–106.

[53] F. Serafino, G. Pio, M. Ceci, Ensemble learning for multi-type classification in heterogeneous networks, IEEE Transactions on Knowledge and Data Engineering 30 (12) (2018) 2326–2339.

[54] I. Kholoshyn, T. Nazarenko, O. Bondarenko, O. Hanchuk, I. Varfolomyeyeva, The application of geographic information systems in schools around the world: a retrospective analysis, Journal of Physics: Conference Series 1840 (1) (2021) 012017.

[55] I. Sabek, M. F. Mokbel, Machine learning meets big spatial data (revised), in: 2021 22nd IEEE International Conference on Mobile Data Management (MDM), 2021, pp. 5–8.

[56] S. Shekhar, P. Zhang, Y. Huang, Spatial data mining, in: Data mining and knowledge discovery handbook, Springer, 2009, pp. 837–854.

[57] P. Stolorz, E. Mesrobian, R. Muntz, J. Santos, E. Shek, J. Yi, C. Mechoso, J. Farrara, Fast spatio-temporal data mining from large geophysical datasets (1995).

[58] S. Shekhar, P. Zhang, S. Chawla, Spatial databases, in: K. Kempf-Leonard (Ed.), Encyclopedia of Social Measurement, Elsevier, New York, 2005, pp. 599–604.

[59] R. C. Souza, R. M. Assunção, D. M. Oliveira, D. B. Neill, W. Meira, Where did i get dengue? detecting spatial clusters of infection risk with social network data, Spatial and Spatio-temporal Epidemiology 29 (2019) 163–175.

[60] T. Nakaya, K. Yano, Visualising crime clusters in a space-time cube: An exploratory data-analysis approach using space-time kernel density estimation and scan statistics, Transactions in GIS 14 (3) (2010) 223–239.

[61] L. Shi, V. P. Janeja, Anomalous window discovery for linear intersecting paths, IEEE Transactions on Knowledge and Data Engineering 23 (12) (2011) 1857–1871.

[62] X. Ye, X. Wei, A multi-dimensional analysis of el niño on twitter: Spatial, social, temporal, and semantic perspectives, ISPRS International Journal of Geo-Information 8 (10) (2019) 436.

[63] R. Medina, G. Hepner, Geospatial analysis of dynamic terrorist networks, in: Values and violence, Springer, 2008, pp. 151–167.

[64] D. Stojanova, M. Ceci, A. Appice, S. Džeroski, Network regression with predictive clustering trees, Data Mining and Knowledge Discovery 25 (2) (2012) 378–413.

[65] C. Desrosiers, G. Karypis, Within-network classification using local structure similarity, in: W. Buntine, M. Grobelnik, D. Mladenić, J. Shawe-Taylor (Eds.), Machine Learning and Knowledge Discovery in Databases, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 260–275.

[66] u. Xie, J. Xu, T.-C. Lu, Automated classification of extremist twitter accounts using content-based and network-based features, in: 2016 IEEE International Conference on Big Data (Big Data), 2016, pp. 2545–2549.

[67] S. Kannan, V. Gurusamy, S. Vijayarani, J. Ilamathi, M. Nithya, S. Kannan, V. Gurusamy, Preprocessing techniques for text mining, International Journal of Computer Science & Communication Networks 5 (1) (2014) 7–16.

[68] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, CoRR abs/1310.4546 (2013). `arXiv:1310.4546`.

[69] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proc. of NAACL-HLT 2019, Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186.

[70] D. E. Rumelhart, J. L. McClelland, C. PDP Research Group (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations, MIT Press, Cambridge, MA, USA, 1986.

[71] S. Shao, C. Tunc, A. Al-Shawi, S. Hariri, One-class classification with deep autoencoder neural networks for author verification in internet relay chat, in: 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA), 2019, pp. 1–8.

[72] X. Glorot, A. Bordes, Y. Bengio, Domain adaptation for large-scale sentiment classification: A deep learning approach, in: ICML'11, Omnipress, Madison, WI, USA, 2011, p. 513–520.

[73] C. Zhou, R. C. Paffenroth, Anomaly detection with robust deep autoencoders, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 665–674.

[74] S. Park, M. Kim, S. Lee, Anomaly detection for http using convolutional autoencoders, IEEE Access 6 (2018) 70884–70901.

[75] S. Šuster, I. Titov, G. Van Noord, Bilingual learning of multi-sense embeddings with discrete autoencoders, arXiv preprint arXiv:1603.09128 (2016).

[76] G. P. Way, C. S. Greene, Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders, in: PACIFIC SYMPOSIUM ON BIOCOMPUTING 2018: Proceedings of the Pacific Symposium, World Scientific, 2018, pp. 80–91.

[77] V. Klema, A. Laub, The singular value decomposition: Its computation and some applications, IEEE Transactions on Automatic Control 25 (2) (1980) 164–176.

[78] A. Maćkiewicz, W. Ratajczak, Principal components analysis (pca), Computers & Geosciences 19 (3) (1993) 303–342.

[79] D. D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.

[80] Y. Wang, H. Yao, S. Zhao, Auto-encoder based dimensionality reduction, Neurocomputing 184 (2016) 232–242, roLoD: Robust Local Descriptors for Computer Vision 2014.

[81] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks (2016). arXiv:1607.00653.

[82] J. Levatic, D. Kocev, M. Ceci, S. Dzeroski, Semi-supervised trees for multi-target regression, Inf. Sci. 450 (2018) 109–127.

[83] L. Breiman, J. Friedman, C. J. Stone, R. A. Olshen, Classification and regression trees, CRC press, 1984.

[84] C. C. Robusto, The cosine-haversine formula, The American Mathematical Monthly 64 (1) (1957) 38–40.

[85] H. Ramchoun, M. A. J. Idrissi, Y. Ghanou, M. Ettaouil, Multilayer perceptron: Architecture optimization and training., Int. J. Interact. Multim. Artif. Intell. 4 (1) (2016) 26–30.

[86] J. T. Hancock, T. M. Khoshgoftaar, Survey on categorical data for neural networks, Journal of Big Data 7 (1) (2020) 1–41.

[87] V. Vovk, The fundamental nature of the log loss function, in: Fields of Logic and Computation II, Springer, 2015, pp. 307–318.

[88] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, D. McClosky, The Stanford CoreNLP Natural Language Processing Toolkit, Proc. of Annual Meeting of the Association for Computational Linguistics: System Demonstrations (2014) 55–60.

[89] G. De Martino, G. Pio, M. Ceci, PRILJ: an efficient two-step method based on embedding and clustering for the identification of regularities in legal case judgments, Artificial Intelligence and Law 30 (3) (2022) 359–390.