World Wide Web
https://doi.org/10.1007/s11280-023-01192-w

HURI: Hybrid user risk identification in social networks

Roberto Corizzo[1] · Gianvito Pio[2,3] · Emanuele Pio Barracchia[2,3] · Antonio Pellicani[2,3] · Nathalie Japkowicz[1] · Michelangelo Ceci[2,3,4]

Received: 16 November 2021 / Revised: 12 April 2023 / Accepted: 26 June 2023

Abstract
The massive adoption of social networks increased the need to analyze users' data and interactions to detect and block the spread of propaganda and harassment behaviors, as well as to prevent actions influencing people towards illegal or immoral activities. In this paper, we propose HURI, a method for social network analysis that accurately classifies users as *safe* or *risky*, according to their behavior in the social network. Specifically, the proposed hybrid approach leverages both the topology of the network of interactions and the semantics of the content shared by users, leading to an accurate classification also in the presence of noisy data, such as users who may appear to be risky due to the topic of their posts, but are actually safe according to their relationships. The strength of the proposed approach relies on the full and simultaneous exploitation of both aspects, giving each of them equal consideration during the combination phase. This characteristic makes HURI different from other approaches that fully consider only a single aspect and graft partial or superficial elements of the other into the first. The achieved performance in the analysis of a real-world Twitter dataset shows that the proposed method offers competitive performance with respect to eight state-of-the-art approaches.

Keywords Social network analysis · Neural networks · Node classification · Risk identification

1 Introduction

In modern society, social networks represent the most common way for millions of users to express their ideas, beliefs and preferences through posts, likes and comments. Such a massive adoption has attracted the interest of many companies and institutions worldwide. Indeed, the analysis of users' interactions and behavior may inspire the design of innovative products and services according to current trends and customer preferences. Other activities

This article belongs to the Topical Collection:*Provided Funding information has to be tagged.*.

✉ Michelangelo Ceci
michelangelo.ceci@uniba.it

Extended author information available on the last page of the article

that can be performed by users of the social network can be the spread of information about important (ongoing or upcoming) events, or the sensitization of people about emerging social causes and issues.

However, since social networks are also considered an innovative and effective tool for propaganda, they can also be used for bad or illegal activities, such as *i)* harassment [1], *ii)* influencing people to adopt illegal practices, *iii)* promoting the use of drugs, or *iv)* spreading religious fundamentalism and political extremism. With regards to the latter, we can also find extreme situations, where terrorist communities exploit social networks, such as Twitter or Facebook, to disseminate their ideas and recruit new people.

Some approaches proposed in the literature rely on the analysis of either the network topology, i.e., the relationships among users [6–9], or the textual content the users post or interact with [10–12]. However, these approaches may be ineffective in the presence of noisy or misleading data. A typical case is that of journalists: They may share many contents related to topics involving words that are usually used in high-risk contexts, such as events related to the use of weapons or explosive devices. This aspect may push methods purely based on the analysis of textual content to erroneously label journalists as risky users (see the left side of Figure 1). On the other hand, journalists tend to be linked to (e.g., through the relationship *follows* in Twitter) users belonging to both safe and risky communities, mainly to stay up-to-date with the latest events. In this respect, methods purely based on the analysis of the network topology (i.e., only considering the users' relationships) will correctly label journalists as safe only if their relationships with other safe users are strongly predominant as compared to those with risky users (see the right side of Figure 1).

In order to overcome the limitations of existing approaches, in this paper, we propose a new method called HURI (Hybrid User Risk Identification in Social Networks) that is able to detect high-risk users in social networks, by analyzing the information conveyed by both the topology of the network and the content posted by the users. This is achieved by a hybrid approach that learns two models (i.e., one for each aspect) that are combined to make the final predictions.

Hybrid approaches have already been proposed in the literature (see Section 2 for an overview). Among them, we can find approaches falling in the relational data mining field, as well as methods for the analysis of heterogeneous networks [13–15], that are able to model

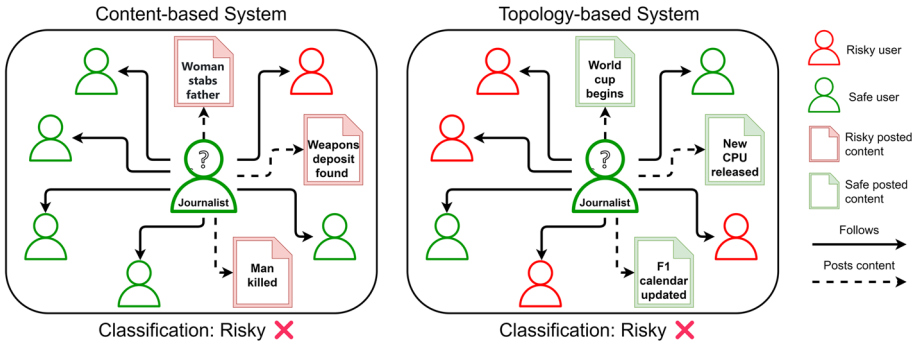

Fig. 1 A graphical representation of common misclassification errors made on a noisy user (e.g., a journalist). On the left, we show a misclassification error commonly made by content-based systems on users who post/interact with apparently risky content, even if he/she is linked with several safe users. On the right, we show a misclassification error commonly made by topology-based systems on users who establish more relationships with risky users than with safe users, even if he/she posts only safe content

input data as entities and relationships. Although this characteristic provides them with the possibility to be applied in multiple domains, they are generally not able to exploit peculiarities exhibited by a specific domain, e.g., the semantics of the content generated by users. Other approaches (e.g., the methods proposed by [16] and [17]) represent one dimension (content or relationships, respectively) and inject additional features summarizing the other (relationships or content, respectively). This strategy usually leads to implicitly provide a higher relevance to one dimension with respect to the other, and to possibly introduce spurious or redundant features. In this context, the challenge we face with the proposed method HURI is represented by the explicit modeling of the semantics of the content generated by the users, as well as their relationships, without introducing spurious features and without assuming a higher relevance of one dimension with respect to the other.While this aspect represents one of the major advantages of HURI over other hybrid approaches, its other key contributions are summarized as follows: *i)* it captures the semantics of the content posted by *risky* and *safe* users, by learning two separate models based on AutoEncoders [18], that are able to represent/embed the users into a numerical feature space; *ii)* it represents the network of relationships established by the users and learns a separate predictive model based on decision trees; *iii)* it properly combines the contribution coming from both dimensions of analysis, by exploiting a stacked neural network that does not consider only the predictions, as usually done by common approaches based on Stacked Generalization [19], but also the confidence about such predictions. The latter provides HURI with the ability of capturing and exploiting the uncertainty of the predictions, making it more robust to the noise in the data.

Experiments conducted on a real-world Twitter dataset show that HURI is able to detect high-risk users more accurately than existing approaches and that is also robust to the presence of noisy data (e.g., journalists).

The rest of the paper is organized as follows. In the next section, we briefly discuss the background and the motivations of the proposed hybrid approach. In Section 3 we review related work in node embedding and classification. In Section 4 we describe the proposed method. In Section 5 we describe the performed experiments and discuss the obtained results. Finally, in Section 6 we draw some conclusions and outline possible future works.

2 Background and motivations

The task solved in this paper falls in the research area of Social Network Analysis (SNA), that is, the study of social structures exploiting network and graph theory [20, 21]. Although SNA has its roots in sociology [22], the concept has evolved over time and is being adopted in multiple fields, such as biology, economics, political science and computer science. Networks studied by SNA consist of *nodes*, that represent, for example, people or organizations, and *edges* between nodes, describing social relationships. Examples of tasks addressed by SNA in the literature include the identification of collaborations between academic co-authors [23], the study of the cohesion among political parties [24], the detection of compromised accounts [25], or the prediction of users involved in criminal incidents [26].

Regarding the last examples, some works in the literature proposed SNA approaches tailored for the detection of propaganda activities about terrorism in social networks. In [27], the authors describe SNA as a tool to fight this problem, and highlight the main tasks investigated in the counter-terrorism field, such as key-player identification [28, 29], community discovery [30, 31], link analysis [32, 33] and dynamic network analysis [34, 35].

The task we solve in this paper falls in the category of *key-player identification*, and aims at identifying, more generally, high-risk users, namely, users who may demonstrate any kind of negative behavior, or exercise a negative influence over the community. Therefore, this task can be considered a particular case of the node classification task in network data. In the literature, we can find several approaches to solve this task, that can be categorized into three main classes, depending on the underlying criteria adopted to define the similarity among users (see Figure 2 for a graphical representation):

- *topology-based:* they consider only the topology of the network of relationships, motivated by the assumption that the similarity among users can be estimated by considering their relationships;
- *content-based:* they focus on the analysis of the content (e.g., posts, comments) generated by users, assuming that similar users will generate or interact with content regarding similar topics;
- *hybrid:* they attempt to combine topology-based and content-based approaches, to exploit the advantages of both viewpoints.

As underlined in Section 1, one major challenge in real-world social networks is the effective identification of high-risk users in the presence of "noisy" data, e.g., safe users who can erroneously be classified as high-risk users when solely considering either the posted content posted or to their relationships. In this case, hybrid approaches should be able to produce more accurate predictions, since they observe two complementary aspects of the social networks. These considerations pushed us towards the design of the hybrid approach proposed in this paper.

3 Related work

In this section, we briefly discuss existing node embedding techniques, that are commonly adopted to represent network nodes in a numerical feature space, as well as existing approaches for node classification.

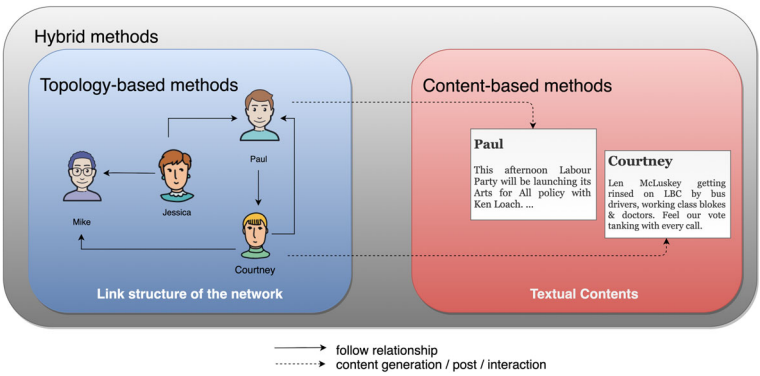

Fig. 2 A graphical representation of topology-based, content-based and hybrid methods for social network analysis

3.1 Node embedding techniques

In the literature many works address the task of node embedding in networks, namely the identification of a numerical feature space for nodes, that embeds the characteristics and the topological role of each node in the network.

Among the most straightforward solutions, we can find dimensionality reduction techniques. In particular, by representing the network of relationships as an adjacency matrix, it is possible to apply methods like Singular Value Decomposition (SVD) [36], Principal Component Analysis (PCA) [37] or Non-negative Matrix Factorizations [38, 39]. Such approaches can identify a reduced, numerical feature space, and deal with data sparsity issues, that are typical of adjacency matrices representing relationships in social networks.

On the other hand, in the literature we can also find methods specifically designed to solve node embedding tasks. For example, *DeepWalk* [40] aims at learning a feature space for nodes that preserves the closeness with their neighboring nodes in the network. The neighborhood of each node is identified by exploiting truncated random walks. A similar approach is adopted by *node2vec* [41]

Other methods, such as *LINE* [42] and *SDNE* [43], perform network embedding using both first-order (i.e., observed links in the network) and second-order (i.e., shared neighborhood among nodes) proximity, with the main goal of preserving both local and global network structure.

A different approach, called *Hashtag2Vec* [44], performs node embedding by exploiting the information conveyed by both the topological structure and the content. The proposed embedding model is able to learn a hashtag representation by optimizing a loss function that takes into account multiple types of relationships: hashtag-hashtag, hashtag-tweet, tweet-word and word-word. However, this method cannot be directly adapted to learn a representation for users, since it explicitly represents and exploits *co-occurrence* relationships among hashtags that cannot be mapped to *friend* or *follow* relationships among users.

3.2 Node classification methods

As mentioned at the beginning of this section, methods for node classification available in the literature can be categorized in three classes: topology-based, content-based and hybrid. Topology-based methods focus on the link structure of the network [45] and exploit it for node classification. A relevant example is the system *GNetMine* [6], that is able to represent arbitrary, also heterogeneous, information networks, and to classify nodes according to their relationships. In general, methods falling into this category are based on collective inference, i.e., they make simultaneous judgements on the same variables of related nodes. In particular, they exploit the so-called relational autocorrelation, a phenomenon that takes place in relational data when the value of a property of one object is highly correlated with the value of the same property of another object [46]. Within this class of approaches, we can find an interesting work [7] that proposes a node-centric framework that exploits only information on the structure of class linkage in the network, that is, only links and class labels. Another work [8] addresses a challenging scenario falling into the within-network classification setting, in partially-labeled networks. Specifically, they combine statistical relational learning and semi-supervised learning to improve the classification performance in sparse networks, by adding "ghost edges" that enable the flow of information from labeled to unlabeled nodes.

The authors of [9] propose an active inference method that learns to identify the cases in which collective classification algorithms make mistakes, and suggests changes to correct

such mistakes. The authors demonstrated that the proposed method outperforms several approaches based on network topology.

In [47], the authors aim to identify Sybil attacks in online social networks, where attackers attempt to carry out harmful actions while posing as (multiple) genuine users. To achieve this goal, the authors exploit the topology of the network, focusing on the strength and on the interactions of the users' relationships. They also incorporate graph-based features, such as betweenness-centrality, to enhance the identification of the attacks.

Focusing on content-based approaches, in [10] the authors propose a method to classify posts and users on Twitter into three different classes: positive, negative and neutral. To this aim, the authors exploit two lexicons, containing, respectively, "positive" and "negative" words. For each tweet in the dataset, a feature vector is constructed, where each feature represents the occurrence of each word (belonging to either the positive or the negative lexicons) in the tweet. The vector is then updated through Word2Vec [48] in order to consider both the semantics and the relationships among words. The obtained features are used to cluster tweets into positive, negative and neutral.

Another content-based approach is Doc2Vec [12], which is an extension of Word2Vec. Its goal is to create a numerical representation of a document, regardless of its length, that can be subsequently exploited by any classification approach based on feature vectors. Contrary to Word2Vec, that extracts semantic vector representations at a word level, Doc2Vec extracts semantic vector representations at a document level, learning distributed representations for both words and documents simultaneously.

A different approach is proposed in [11], which goal is to detect the presence of cyber-terrorism and extremism contents in textual data. Together with classical weighting methods, like TF-IDF and binary weighting, the authors propose a novel "fuzzy set-based weighting method" that appears to be more appropriate for the specific task.

In [49], the authors present a study on keyword-based indicators and discusses their effectiveness in highlighting frustration and discrimination, and in estimating the risk of radicalization for users of the social network.

The work in [16] focuses on the analysis of a network in which nodes represent tweets, while edges represent hashtags and mentions. The authors show that the adoption of relational probability trees, with features built from both the content and the structure of the network, leads to an accurate user classification. However, social relationships such as *friend* or *follows*, are not explicitly taken into account.

Shifting the focus on hybrid approaches, in [17] the authors propose a method based on Adaboost to analyze both content-based and topology-based features, to automatically detect extremist accounts on Twitter. The considered features include hashtags, the tokens included in hashtags, the harmonic closeness between a target node and the set of known ISIS supporter nodes, and the expected hitting time of a random walk from individual nodes to known ISIS nodes. However, like [16], this approach does not explicitly take into account the network of relationships, but only use centrality measures.

In [50], the authors analyze a real-world dataset extracted from Instagram to identify *influential users* who may contribute to the dissemination of harmful information by advertising specific products. They focus on the content rather than the network structure and exploit a combination of high-level features extracted from images such as color scheme, semantics, and advertising aspects. In the experiments, the authors compare their system with a previous study that solely used the textual content and prove that their image-based method is more accurate.

In [51], the authors propose an optimization tool that exploits both the content and the topology of social networks. The authors show that the information conveyed by the topology

of the network is usually noisy, and aim to support such a dimension of analysis with the content associated with the nodes. Although the authors considered a different task (i.e., community detection) with respect to that we solve in this paper, they analogously proved that the combined exploitation of content and topology provides better results than those achieved considering only the network topology.

The structure of the network of relationships is fully exploited also by methods working on heterogeneous information networks. Among such methods, it is worth mentioning *HENPC* [13], that solves the multi-type node classification task by extracting overlapping and hierarchically organized clusters, that are subsequently used for predictive purposes. Analogously, the method *Mr-SBC* [14], as well as its multi-type counterpart *MT-MrSBC* [15], adopts the naïve Bayes classification method for the multi-relational network setting, thus allowing the consideration of both the content and the relationships among the involved entities.

One common limitation of the approaches previously mentioned is that the content of the posts is represented implicitly or indirectly, that is, through the relationships between words and posts, and between posts and users, without exploiting the semantics of the textual content. Contrary to other *hybrid* approaches [14, 15, 52], HURI is able to explicitly take into account the semantics of the content generated by users, and their role in the network. This is not limited to including topological features together with those depending on the content (as performed by other methods), but it consists in explicitly exploiting the network of relationships as complementary information.

4 The proposed method HURI

Before describing the proposed method HURI, we briefly formalize the task we are solving. In particular, HURI analyzes a network $G = \langle N, E_N, C, E_C \rangle$, where:

- $N_L = N_L^{(s)} \cup N_L^{(r)}$ is the set of nodes representing users whose label is known, where $N_L^{(s)}$ is the set of *safe* users (i.e., with label *S*) and $N_L^{(r)}$ is the set of *risky* users (i.e., with label *R*);
- N_U is the set of nodes representing unlabeled users;
- $N = N_L \cup N_U$ is the set of nodes representing all the users (either labeled or unlabeled);
- $E_N \subseteq N \times N$ represents a relationship (e.g., *follower*) between users;
- C is the set of textual documents;
- $E_C \subseteq N \times C$ represents the relationships among users and textual contents, namely, that a given user generated/posted (or interacted with) a given textual content.

The task solved by our method is the estimation of the risk and the prediction of the corresponding label for the users in N_U. This means that our approach works in the *within-network* (or *semi-supervised transductive*) setting [53]: nodes for which the label is known are linked to nodes for which the label must be predicted (see [13, 54]). This setting differs from the *across-network* (or *semi-supervised inductive*) setting, where learning is performed from one (fully labeled) network and prediction is performed on a separate, presumably similar, unlabeled network (see [55, 56]). This provides our method with a significant advantage, since it can fully exploit the textual content and the relationships of unlabeled users during the training phase.

The general workflow of the proposed method consists of three phases (see Figure 3): *i)* network topology analysis, *ii)* semantic content analysis and *iii)* their combination.In particular, we learn a predictive model based on a set of features that represents each user

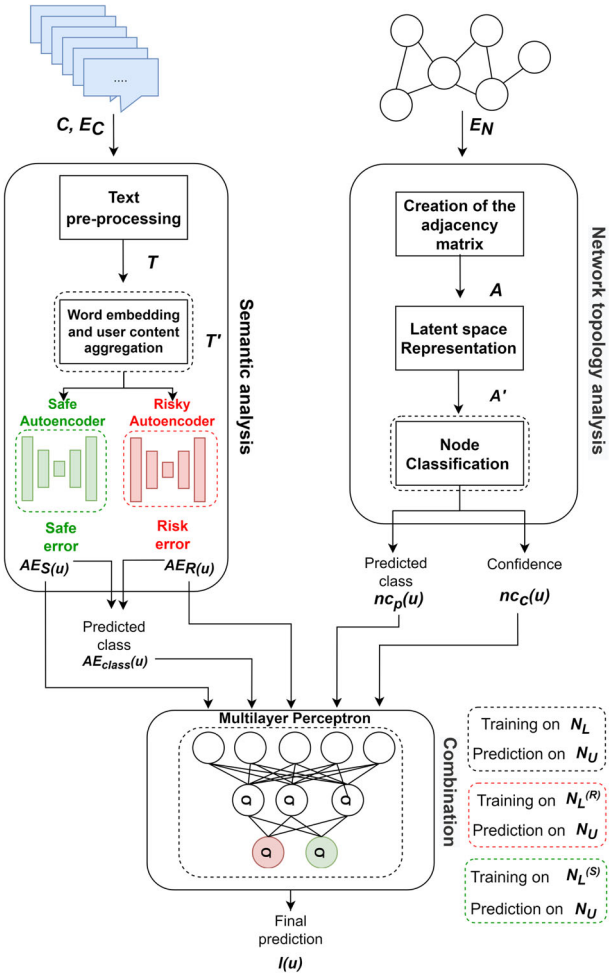

Fig. 3 General workflow of the proposed method

on the basis of her/his relationships with other users (e.g., follows), and a predictive model based on the textual content she/he posted. Finally, we leverage the output of such models in combination to obtain a final model that is less prone to return incorrect classifications, that may possibly derive from the partial analysis of each single aspect in isolation. The adopted combination approach is inspired by the Stacked Generalization framework [19], which aims to reduce the bias of each single task. However, in addition to the classical Stacked Generalization approach, we also exploit the degree of confidence of the returned predictions, making HURI more robust to the possible uncertainty exhibited by the models learned separately from the textual content and from the network of relationships.

Although other methods take into account both aspects (see Section 3.2), they are not able to simultaneously exploit their full potential. In particular, either *i)* they include simple topological features together with those related to the content (see the work by [16, 17]), or

ii) although they are able to explicitly represent both the topology and the content (see the work by [13–15]), specific peculiarities of textual content (e.g., the semantics) or network relationships are not taken into account.

4.1 Network topology analysis

The goal of phase is to exploit the network structure, i.e., the relationships in which users are involved, for predictive purposes. The most straightforward approach consists in training a prediction model directly from an adjacency matrix built from the network of relationships. In particular, given the network $G = \langle N, E_N, C, E_C \rangle$, and considering n_i as the i-th user of the network, the adjacency matrix $A \in \{0, 1\}^{|N| \times |N|}$, can be easily constructed by setting $A_{ij} = 1$ if $(n_i, n_j) \in E_N$; $A_{ij} = 0$ otherwise.

However, social networks are usually not densely connected, leading to the construction of highly sparse adjacency matrices. For example, according to the financial results reported in Q2 2019 IR Statement[1], on Facebook there were 1.59 billion active daily users in June 2019. Assuming an average number of 1000 friends per user, the sparseness of an adjacency matrix representing the Facebook network would be more than 99.99%.

To deal with this issue, in the literature, we can find several dimensionality reduction techniques, such as SVD [36], PCA [37] and NMF [38], that aim at identifying a new, reduced, feature space, with a lower sparseness rate. Such a task can also be performed by exploiting embedding techniques, such as *AutoEncoder bottleneck encodings* [57] or *Node2vec* [58].

We do not bind HURI to a specific approach, but we allow the adoption of any solution that is able to reduce the dimensionality of the adjacency matrix. Indeed, in our experiments (see Section 5), we evaluated the performance exhibited by HURI with different solutions to solve this step.

Formally, given the adjacency matrix A, the adoption of a dimensionality reduction technique leads to a new matrix $A' \in \mathbb{R}^{|N| \times k_t}$, where k_t is the desired dimensionality of the reduced feature space for the topology analysis. Once a compact, dense representation of the network has been identified, we train a node classification model nc on the set of labeled users N_L. This model is then exploited to predict the label for all the unlabeled users N_U as either S (*safe*) or R (*risky*).

Specifically, we require the classification model nc to be able to produce a pair $\langle nc_p(u), nc_c(u) \rangle$ for each user $u \in N_U$, where $nc_p(u)$ represents the predicted label and $nc_c(u)$ represents the confidence of the prediction. This is fundamental in order to provide the final combination step (see Section 4.3) with complete information about the prediction.

In HURI, we specifically rely on tree-based classifiers for this purpose. This choice is mainly motivated by the state-of-the-art performances exhibited by such approaches in semi-supervised settings [59], and specifically on network data [56]. The learned decision trees consist of nodes and branches, identified through a top-down induction procedure that recursively partitions the set of training examples. Each partitioning criterion, also called *split*, is based on a feature on a value/threshold, which are greedily determined by maximizing some heuristics.

In HURI we maximize the reduction of the classical *Gini Index* [60], that is based on the purity of each class measured after the split. More formally, the Gini Index is defined as:

$$Gini(n) = 1 - (p_r^2 + p_s^2),$$

[1] https://investor.fb.com/investor-news/press-release-details/2019/Facebook-Reports-Second-Quarter-2019-Results/default.aspx

where p_s and p_r are the relative frequencies of safe and risky users in the tree node n, respectively.

Given an unlabeled user $u \in N_U$, the decision tree built by HURI returns the predicted label $nc_p(u)$ as the majority class in the leaf node in which u falls in the learned tree, and the confidence value $nc_c(u)$, which corresponds to the purity of such a leaf computed according to the examples falling in such a leaf during the training phase.

A graphical overview of the topological analysis performed by our method can be seen in the bottom section of Figure 3.

4.2 Semantic content analysis

As already mentioned in Section 1, together with the users' relationships, we leverage the textual content that users interacted with (e.g., posted and commented on). In particular, given the network $G = \langle N, E_N, C, E_C \rangle$ as formalized in Section 4, we first pre-process the textual content in C, using a standard Natural Language Processing pipeline, consisting of tokenization, stopword removal, stemming and metadata removal [61]. According to E_C, we associate each user in N with a representation that depends on his/her textual content. In particular, we build the dataset T by concatenating the textual content of all the documents associated with each user, according to their timestamp. This approach has two advantages: *i)* the documents of each user are not considered independently, but in a combined form; *ii)* the temporal evolution of the topics discussed in different documents can be exploited in the definition of the context. Then we train a *Word2Vec* model [48] from all the textual documents associated with the labeled users N_L and exploit it to process the dataset T. In particular, using this model, we obtain a k_c-dimensional numerical vector (embedding) for each word, that represents its semantic meaning. We then use it to associate a k_c-dimensional numerical vector to each user, according to all the terms appearing in the textual content the user interacted with.

Formally, let $words(u)$ be the list of words appearing in the textual content the user u interacted with, and $w2v(w)$ the embedding generated by *Word2Vec* for the word w. We exploit the "additive compositionality" property of word embeddings [48], according to which, not only similar words appear close to each other in the feature space, but the sum of vectors in the embedding space resembles an "AND" concatenation. As a result, if two sentences appear in the same context, their vectors obtained as the sum of word embedding vectors will still be close to each other according to a similarity measure. Analogously, in our case, two users whose vectors have been obtained by the sum of word embedding vectors appearing in their documents will be close/similar to each other. Formally, we compute the semantic vector representation $sem(u)$ for each user $u \in N$ as follows:

$$sem(u) = \sum_{w \in words(u)} w2v(w) \tag{1}$$

In this way, we obtain a new dataset $T' \in \mathbb{R}^{|N| \times k_c}$, consisting of the semantic vector representation for all the users in N.

Since, in this specific context, it is expected that the textual contents are strongly polarized towards the label *safe*, whereas data for the label *risky* would be generally scarce, we adopt AutoEncoders [18] to effectively model the different data distributions of the two classes. AutoEncoders work by compressing input data into a latent-space representation and then reconstructing the output from this representation. This characteristic has been exploited in the literature to perform anomaly detection and classification [62–64], relying on the

analysis of the reconstruction error. For the classification task, the most effective approach consists in training a one-class AutoEncoder model for each possible label and assessing its reconstruction capability on unseen data. If a high reconstruction error of the AutoEncoder is observed, then the given object most likely belongs to a different class than that assumed by training data instances. This solution is preferred with respect to standard multi-class classifiers, since, as observed by [65], the performance of one-class classifiers appears more stable with respect to the level of class imbalancing.

Following such an approach, starting from the dataset T', we build two AutoEncoders, i.e., one for the label *S (safe)* and one for the label *R (risky)*. More formally, an AutoEncoder aims at learning two functions: the encoding function $enc : \mathcal{X} \rightarrow \mathcal{F}$ and the decoding function $dec : \mathcal{F} \rightarrow \mathcal{X}$, such that:

$$\langle enc(\cdot), dec(\cdot) \rangle = \underset{\langle enc(\cdot), dec(\cdot) \rangle}{\operatorname{argmin}} \|T' - dec(enc(T'))\|^2, \tag{2}$$

where $\mathcal{X}$ is the data input space of T' (i.e., $\mathcal{X} = \mathbb{R}^{k_c}$), and $\mathcal{F}$ is the encoding space learned by the AutoEncoder.

The functions $enc(\cdot)$ and $dec(\cdot)$ should be parametric and differentiable with respect to a distance function, so that their parameters can be optimized by minimizing the reconstruction loss.

The architecture of an AutoEncoder consists of one or more hidden layers, where the output of the i-th hidden layer represents the i-th encoding level of the input data. The last layer of the AutoEncoder is of the same size of the input layer and aims to return the reconstructed input representation after the decoding stage. In this work, we adopt two hidden layers for the encoding stage and two hidden layers for the decoding stage (see Figure 4 for a graphical representation of the architecture).

Without loss of generality, in the following we briefly explain how an AutoEncoder with one hidden layer works. The formalization can then be easily extended to AutoEncoders with multiple hidden layers. In particular, the encoding stage takes the input $sem(u) \in \mathbb{R}^{k_c} = \mathcal{X}$ and maps it to an hidden representation $z(u) \in \mathbb{R}^{k_c/2} = \mathcal{F}$. Formally:

$$z(u) = \sigma(\mathbf{W} \cdot sem(u) + b) \tag{3}$$

where σ is a sigmoid activation function, $\mathbf{W}$ is a weight matrix, and b is a bias vector, all associated to the encoding part.

The decoding stage reconstructs $sem(u)$ from z as:

$$sem'(u) = \sigma'(\mathbf{W}' \cdot z + b') \tag{4}$$

where σ' is a sigmoid activation function, $\mathbf{W}'$ is a weight matrix, and b' is a bias vector, all associated to the decoding part.

The process aims at minimizing the following reconstruction loss:

$$\begin{aligned} &\phi(sem(u), sem'(u)) = \|sem(u) - sem(u)'\|^2 = \\ &\|sem(u) - \sigma'(\mathbf{W}'(\sigma(\mathbf{W} \cdot sem(u) + b)) + b')\|^2 \end{aligned} \tag{5}$$

The learning of $\mathbf{W}$, $\mathbf{W}'$, b, b' takes place according to the minimization of the reconstruction loss ϕ on training data, which computes the difference between the original and reconstructed versions.

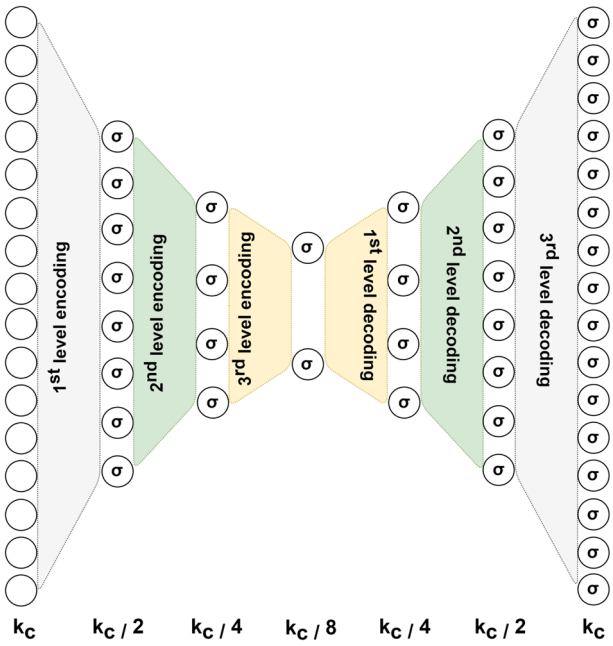

Fig. 4 A graphical representation of the proposed AutoEncoder architecture for semantic content analysis: Three stages of encoding and decoding, that aggregate and reconstruct the aggregated semantic representation of each user. The lowest reconstruction error obtained between the two AutoEncoders (one for the *risky* label and one for the *safe* label) is used to perform a content-based classification of the user

As previously stated, we build two different AutoEncoders, namely, AE_R, trained from the semantic vector representation of known risky users $N_L^{(r)}$, and AE_S, trained from the semantic vector representation of known safe users $N_L^{(s)}$.

Given an unlabeled user u, we feed both the AutoEncoders AE_S and AE_R with his/her semantic representation $sem(u)$, in order to compute the reconstruction errors, $AE_S(u)$ and $AE_R(u)$ respectively, according to the function $\phi(\cdot, \cdot)$. Therefore, the output of the semantic analysis for a user $u \in N_U$ is threefold:

- the reconstruction error $AE_S(u)$ computed by the AutoEncoder AE_S;
- the reconstruction error $AE_R(u)$ computed by the AutoEncoder AE_R;
- the predicted label $AE_{label}(u) \in \{S, R\}$ (safe or risky), according to the minimum error measured by the AutoEncoders AE_S and AE_R.

More formally, $AE_{label}(u)$ is computed as follows:

$$AE_S(u) = \phi(AE_{S_{dec}}(AE_{S_{enc}}(sem(u)), sem(u)) \tag{6}$$

$$AE_R(u) = \phi(AE_{R_{dec}}(AE_{R_{enc}}(sem(u)), sem(u)) \tag{7}$$

$$AE_{label}(u) = \underset{x \in \{R,S\}}{\operatorname{argmin}}\{AE_x(u)\} \tag{8}$$

We stress that the adopted strategy allows us to catch and focus on the semantics of the textual contents, and to properly model safe and risky users accordingly, without introducing spurious features based on topological characteristics of the network (as done, for example,

by [16, 17]). Topological aspects, on the contrary, are specifically considered by the phase described in the previous subsection.

A graphical view of this phase can be seen in the left section of Figure 3.

4.3 Combining topology and semantics in textual contents

The final step aims at estimating the final risk score to assign to the unlabeled users N_U. This problem is solved by learning a model able to combine the outputs of the network topology analysis (predicted class nc_p and prediction confidence nc_c) and semantic analysis (safe error AE_S, risky error AE_R and label AE_{label}).

Methodologically, we exploit a Multi-Layer Perceptron (MLP) [66] in a stacked generalization setting [19]. An MLP is an Artificial Neural Network, consisting of an input layer that receives the signal, an output layer that produces the output (i.e., the predicted label) and (possibly) multiple hidden layers. More formally, the predicted label $l(u)$ for the user u is obtained as:

$$l(u) = MLP(nc_p(u), nc_c(u), AE_S(u), AE_R(u), AE_{label}(u)) \tag{9}$$

The architecture of the MLP adopted in this work is shown at the bottom of Figure 3 and consists of the following layers.

The **input layer** consists of 5 neurons and receives the values of $nc_p(u)$, $nc_c(u)$, $AE_S(u)$, $AE_R(u)$ and $AE_{label}(u)$.

The **hidden layer** consists of 3 neurons that use *sigmoid* as the activation function. The adoption of the sigmoid function is motivated by its ability to extract non-linear dependencies between input and output values [67], whereas the number of neurons for the hidden layer is heuristically defined between the number of input and output neurons [68].

The **output layer** consists of 2 neurons that exploit the softmax activation function. This choice became highly popular in classification problems, due to its advantage to return the probability for each class and predict the class according to the highest probability. For this purpose, the class attribute for training examples is subject to one-hot-encoding [69], which leads to two binary class attributes, only one of which assumes a value of 1. According to this setting, the first neuron returns the probability that the user is *safe*, whereas the second neuron returns the probability that he/she is *risky*. The highest probability is chosen to make the final prediction.

In this architecture, our MLP model acts as a stacking meta-model that learns how to effectively combine the predictions of the different analytical steps, thus maximizing the overall predictive accuracy. As previously emphasized, this last step allows us to automatically catch both the aspects (topology and semantics), without imposing any user-defined criteria. Moreover, this approach is smarter than simple averaging approaches (or variants based on majority voting), since it can exploit possible patterns in the output provided by the other two phases as well as additional features, like the confidence and the reconstruction errors.

5 Experiments

In this section, we describe the experiments we performed to evaluate the performance achieved by HURI. Before presenting the results, in the following subsections, we briefly describe the datasets, the considered competitor systems and the experimental setting.

5.1 Datasets

In this work, we exploit a real-world Twitter dataset[2], collected using a crawling system compliant with the Twitter policies, and the Conditional Independence Coupling (CIC) algorithm to obtain a representative sample of users, with no specific hashtag. The sample produced by CIC is mathematically proven to converge to the stationary distribution of the population. CIC also allows to constrain the sampling process to a desired geographic location, on the basis of geo-location information and self-reported location. In our dataset, the geographic location is restricted to users who are resident in the United States [70]. Each tweet is associated to a sentiment value, i.e. an integer value which represents the polarity of the message, computed through Stanford CoreNLP Toolkit [71], and manually revised by 3 domain experts.

The ground truth for users (i.e., *risky (R)* or *safe (S)*) has been defined using two different strategies, leading to the construction of two different datasets:

- **Keywords**. We consider a tweet as *risky* if it contains at least one keyword included in two specific manually-curated lists. The first is related to terrorism and threats[3], whereas the second contains keywords related to hate against immigrants and women[4]. We assign a score to each user computed as the ratio between the number of her risky tweets and the number of her tweets. This strategy assumes 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, calculated as the sum of the sentiment score of their tweets, that was already pre-computed in the original dataset through the CoreNLP toolkit. This strategy assumes that users who post multiple tweets with a negative sentiment are more likely to be *risky*.

After sorting the users according to their score, three expert reviewers performed a manual inspection of their tweets, that led to select the *safest* (from the top of the list) and the *riskiest* (from the bottom of the list) users. This selection allowed us to ensure the correctness of the user labeling procedure, avoiding incorrect labels in the ground truth (more likely occurring for users with intermediate scores) that would have led to misleading conclusions in the evaluation. An additional step was carried out to inject noisy data under controlled conditions. Specifically, we defined *borderline* users who, in this case, may correspond to the journalists who possibly share negative textual contents for informative purposes, but are primarily connected with *safe* users. For this purpose, 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 showing no connection with other users.

This process led to defining a dataset of 1467 safe users (including 263 borderline users) and 1470 risky users, described by 7,686,231 tweets, for the strategy based on the keywords, and a dataset of 2241 safe users (including 304 borderline users) and 1033 risky users, described by 10,016,749 tweets, for the strategy based on the sentiment.

5.2 Competitor methods

We ran the proposed method HURI using different approaches for the dimensionality reduction of the topological analysis, namely SVD [36], PCA [37], NMF [38], *AutoEncoder*

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

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

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

bottleneck encodings [57] and *Node2vec* [58]. Moreover, we considered different values for k_t and k_c, i.e., $k_t \in \{128, 256, 512\}$ and $k_c \in \{128, 256, 512\}$. The textual content was pre-processed (i.e., we removed metadata, retweets, mentions and URLs) and exploited to obtain the semantic vector representation $\mathbf{sem}(\mathbf{u})$ for each user $u \in N$.

Regarding model hyperparameters for all the neural network-based architectures (i.e., the autoencoders for the semantic content analysis, the autoencoder for the network topology analysis, and the MLP for the combination phase), we followed the heuristics proposed by [72]. Specifically, we initially experimented with different configurations for $learning\ rate$ (negative powers of 10, starting from a default value of 0.01) and $batch\ size$ (powers of 2) using a 20% validation set. Preliminary results suggested that the different configurations did not affect performance metrics significantly. For this reason, the final experiments were performed with the following model configuration: $epochs = 500$, $learning\ rate = 0.0001$, $batch\ size = 32$.

The results obtained were compared with those achieved by eight competitor approaches, each belonging to a different category (topology-based, content-based and hybrid), namely:

- *GNetMine* [6], that is a topology-based method able to classify unlabeled nodes organized in (also heterogeneous) information networks. This comparison allows us to evaluate the performance of HURI against a state-of-the-art method that properly models users' relationships.
- *Doc2Vec* [12], that creates a numerical representation of a document. We apply the method to the textual content and exploit the embedding vector for classification. In the experiments, we consider different values for the dimensionality of the Doc2Vec embedding vector, namely, 128, 256, and 512. As for the downstream classifier, we adopted three different methods, namely:

  – Support Vector Machines implemented in *scikit-learn* [73], with the RBF Kernel, with the adjustment of class weights to compensate the class imbalance.
  – Random Forests implemented in *scikit-learn* [73], with the adjustment of class weights to compensate the class unbalancing. We set the number of trees equal to 100, the minimum number of examples per leaf to 2 and adopted the Minimal Cost-Complexity Pruning, considering the optimal value of the α parameter in $\{0.0, 0.2, 0.5, 1.0, 2.0\}$.
  – Multi Layer Perceptron (MLP), designed with an input layer whose size depends on the Doc2Vec embedding vector size (128, 256, or 512 neurons), an hidden layer with 128 neurons (corresponding to 100%, 50%, or 25% of the input features based on the Doc2Vec embedding vector size) using the sigmoid activation function, and an output layer with 2 neurons with softmax activations.

These two systems represent state-of-the-art methods able to properly model the content generated by users. They are able to catch the semantic content thanks to the adoption of *Doc2Vec* as an embedding approach.

As for hybrid methods, we consider the following approaches:

- *Doc2Vec + Node2Vec* [12], to synergically extract embedding vector representations for both the textual content (Doc2Vec) and the topology component (Node2Vec) of the data. The same experimental setting used for Doc2Vec is adopted in this approach. Specifically, we extract embedding vectors of dimensionality 128, 256, and 512, for both the textual content and the topology content. The two embedding vectors are concatenated and provided as inputs to Support Vector Machines, Random Forest, or Multi-Layer Perceptron, which are adopted as base models for the downstream classification task.

- *MrSBC* [14], that is a state-of-the-art relational classification method, based on a combination of the naïve Bayes classification framework and first-order rules, able to work on data stored on a relational database. The database schema defined for the system consists of: *i)* the *users* table, containing the user IDs and their label; *ii)* the *users_users* table, containing pairs of user IDs that represent their relationships in the network; *iii)* the *users_posts* table, that contains the ID of tweets, each associated to the user who posted it and to the sentiment score; *iv)* the *posts_words* table, that represents the words contained in each tweet. In the experiments, we considered different values of its parameter *max_length_path*, i.e., the maximum length of the paths considered in the exploration of the relational schema. In particular, we evaluated the results with $max\_length\_path \in \{3, 4, 5, 6\}$.
  These methods are capable of considering both the content posted by the users and the network topology, like the proposed method HURI. Therefore, they allow us to directly compare HURI to state-of-the-art hybrid approaches.

5.3 Experimental setup

Our experiments were carried out according to a stratified 5-fold cross validation scheme, that subdivides users randomly into 5 different folds and alternatively considers users in one fold as N_U and users in the remaining 4 folds as N_L. The stratified approach preserves the ratio of safe and risky users. For the evaluation, the workflow shown in Figure 3 was repeated once for each fold and the results obtained were averaged.

The metrics used for the evaluation of the performance achieved by the different methods are *precision*, *recall*, *F1-Score* and *accuracy*, where TP (True Positive), TN (True Negative), FP (False Positive) and FN (False Negative) were computed by considering R (risky) as the positive class.

We report separate metric values for *All Users* and *Borderline* users. This choice is important to provide a dual perspective on our quantitative assessment: a more general one (all users), and a more specific one that focuses on users at the boundary between safe and risky users, who may be more challenging to classify (borderline users).

We recall that, in our study, borderline users are not harmful to the community, but share textual contents on sensitive/risky topics with informative purposes, resembling risky users. For borderline users, we only collect the accuracy, since we assume they are all safe users, with the result that the accuracy corresponds to the recall of the safe class, and it is not correct to compute the precision. The stratified random sampling that we performed also aimed to preserve the ratio of *borderline* users within the set of *safe* users.

5.4 Results and discussion

The first aspect that we discuss concerns the role of the parameters k_t and k_c and the sensitivity of predictive performances of HURI to their values. In Tables 1 and 2, we report the results obtained by HURI with the best configurations of k_t and k_c, which are: $\langle k_t = 128, k_c = 256\rangle$; $\langle k_t = 128, k_c = 512\rangle$; $\langle k_t = 256, k_c = 128\rangle$; $\langle k_t = 512, k_c = 128\rangle$. From the results, it is possible to observe that the configuration $\langle k_t = 128, k_c = 256\rangle$ offers the best trade-off for the two tasks: the discrimination between risky and safe users (F1-Score on all the users) and the correct classification of *borderline* users as safe users (accuracy on borderline users). Such a result gives a clear idea of the need to use larger vectors for the representation of the content than for the network structure, where 128-sized vectors appear to be enough. For

Table 1 Average performance on the Keywords dataset

HURI (kt = 128, kc = 256)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.649	0.876	0.733	0.668	0.442
SVD	0.695	0.859	0.761	0.721	0.581
Node2Vec	0.660	0.755	0.694	0.666	0.574
Adjacency Matrix	0.925	0.399	0.529	0.681	**0.868**
PCA	0.795	0.824	**0.808**	0.803	0.630
HURI (kt = 128, kc = 512)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.709	0.797	**0.751**	0.735	0.766
SVD	0.690	0.791	0.737	0.718	0.736
Node2Vec	0.679	0.679	0.679	0.678	0.679
Adjacency Matrix	0.906	0.398	0.515	0.673	**0.830**
PCA	0.694	0.778	0.734	0.717	0.766
HURI (kt = 256, kc = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.679	0.763	0.718	0.700	0.781
SVD	0.690	0.786	**0.735**	0.716	0.762
Node2Vec	0.709	0.697	0.702	0.705	0.694
Adjacency Matrix	0.928	0.290	0.441	0.633	**0.947**
PCA	0.691	0.776	0.731	0.714	0.774
HURI (kt = 512, kc = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.686	0.786	0.733	0.713	0.770
SVD	0.687	0.788	**0.734**	0.714	0.736
Node2Vec	0.694	0.710	0.701	0.698	0.683
Adjacency Matrix	0.926	0.291	0.443	0.634	**0.943**
PCA	0.681	0.774	0.725	0.706	0.777
Best Competitors					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
MrSBC (max_length=6)	0.500	1.000	**0.667**	0.500	0.000
D2V+N2V+SVM (kt=128, kc=512)	0.729	0.398	0.513	0.575	**0.684**

The best results for each configuration in terms of F1-Score and accuracy of classification of borderline users are highlighted in bold

Table 2 Average performance on the Sentiment dataset

HURI (kt = 128, kc = 256)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.719	0.560	0.628	0.796	0.885
SVD	0.748	0.575	**0.649**	0.808	0.845
Node2Vec	0.622	0.666	0.643	0.772	0.739
Adjacency Matrix	0.861	0.320	0.466	0.774	**0.928**
PCA	0.723	0.588	0.648	0.803	0.840
HURI (kt = 128, kc = 512)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.430	0.337	0.376	0.754	0.928
SVD	0.446	0.362	**0.399**	0.764	0.896
Node2Vec	0.358	0.371	0.364	0.728	0.837
Adjacency Matrix	0.513	0.203	0.291	0.743	**0.949**
PCA	0.428	0.358	0.390	0.757	0.896
HURI (kt = 256, kc = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.381	0.296	0.332	0.681	0.843
SVD	0.578	0.451	**0.506**	0.777	0.880
Node2Vec	0.478	0.494	0.485	0.740	0.811
Adjacency Matrix	0.637	0.323	0.407	0.757	0.829
PCA	0.434	0.358	0.392	0.759	**0.893**
HURI (kt = 512, kc = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.363	0.297	0.327	0.673	0.843
SVD	0.608	0.449	0.515	0.785	0.883
Node2Vec	0.467	0.516	0.488	0.736	0.677
Adjacency Matrix	0.659	0.313	0.414	0.765	0.869
PCA	0.580	0.471	**0.519**	0.781	**0.901**
Best Competitors					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
MrSBC (max_length=6)	0.310	1.000	**0.473**	0.313	0.000
GNetMine	0.388	0.195	0.247	0.694	**0.898**

The best results for each configuration in terms of F1-Score and accuracy of classification of borderline users are highlighted in bold

the correct classification of *borderline* users, it is apparently necessary to further extend the feature space for the representation of the content.

Other configurations, different from $\langle k_t = 128, k_c = 256 \rangle$, can lead to a higher accuracy on *borderline* users, but lead to a lower overall F1-Score for the task of discriminating between risky and safe users. This phenomenon is evident for the Keywords dataset (Table 1), where the best configuration achieves an F1-Score on all users of 0.808 and an accuracy on *borderline* users of 0.868, whereas the other configurations achieve an inferior performance in both aspects (see $\langle k_t = 128, k_c = 512 \rangle$, which achieves in the best case an F1-Score on all users of 0.751 and an accuracy on *borderline* users of 0.830), or obtain high values of accuracy on *borderline* users at the cost of a drastic reduction of the F1-Score on all the users (see $\langle k_t = 256, k_c = 128 \rangle$, which yields in the best case an F1-Score on all users of 0.735 and an accuracy on *borderline* users of 0.947, and $\langle k_t = 512, k_c = 128 \rangle$, which provides an F1-Score on all users of 0.734 and an accuracy on *borderline* users of 0.943). Similarly, on the Sentiment dataset (Table 2), the best configuration achieves an F1-Score on all users of 0.649 and an accuracy on *borderline* users of 0.928, whereas the other configurations globally exhibit lower performances (see $\langle k_t = 256, k_c = 128 \rangle$, which achieves in the best case an F1-Score on all users of 0.506 and an accuracy on *borderline* users of 0.893, and $\langle k_t = 512, k_c = 128 \rangle$, which yields in the best case an F1-Score on all users of 0.519 and an accuracy on *borderline* users of 0.901) or provide just a small improvement to the accuracy on *borderline* users, resulting in an excessive penalization of the overall F1-Score on all the users (as seen in $\langle k_t = 128, k_c = 512 \rangle$, which obtains an F1-Score on all users of 0.399 and an accuracy on *borderline* users of 0.949).

Comparing the results in terms of network representation for HURI, we can observe that, except for some specific cases (see $\langle k_t = 256, k_c = 128 \rangle$ and $\langle k_t = 512, k_c = 128 \rangle$ for the dataset based on sentiment), considering the full adjacency matrix led to the best result on the task of classifying *borderline* users as *safe* users. This is an expected behavior since they have been defined in the dataset according to their relationships, that are fully and explicitly represented by the adjacency matrix. However, as regards the F1-Score on the whole dataset, we can observe that SVD and PCA led to the best results, without significantly affecting the accuracy on *borderline* users. This means that they were able to effectively represent the general network structure, leading to a better generalization of the learned model and good overall robustness to the presence of noise (i.e., journalists). Overall, HURI leads to satisfactory results with any of the considered approaches for the representation of the network, except for the AutoEncoder that in some cases leads to significantly lower results. This means that *i)* AutoEncoder performs worse, compared to statistical approaches like PCA and SVD, when adopted to identify a representation of strongly sparse networks and leads to underfitting; *ii)* HURI is generally able to correctly balance the contribution from the fusion of the information in the content posted by users with that in the network structure, and effectively discriminate between safe and risky users. These conclusions apply to both the considered datasets and show that HURI is a suitable solution to analyze domains characterized by heterogeneous and noisy data, structured as a network, offering better generalization and robustness capabilities than other methods.

In order to further assess the validity of our work, we perform an ablation study that aims to ascertain that all components in the proposed method HURI provide a positive contribution, which translates into an improvement in terms of classification accuracy. Specifically, we allow HURI to analyze the textual content or the relationships among users in isolation, devising two simplified variants of the method that differ in the combination stage, namely:

- **HURI only content**: The MLP adopted for combining the contribution of the two perspectives is trained without considering the predicted class and the confidence returned by the component for the network topology analysis. Instead, for the latter, during the training the ground truth of the data is used as the label, while the confidence is set to 1.0. During the prediction phase, the majority class of the training set is considered, with a confidence factor set to the ratio between the number of majority class samples and the number of samples in the training set. The rationale is to provide the method with most reliable source of information for the training stage, without incorporating the potential smoothness introduced by the confidence factor, which is exploited by the full version of HURI.
- **HURI only relationships**: The MLP adopted for combining the contribution of the two perspectives is trained without considering the actual reconstruction error predicted by the Autoencoder models, which represent the semantic analysis component of HURI. Instead, the reconstruction error for the true label is replaced such that it is lower than the reconstruction error of the opposite label. During the prediction, the predicted class is set to the majority class of the training set, while the safe and the risky errors are set such that the one of the majority class is lower than that of the minority class. The rationale is to provide the model with reasonable and informative input for the textual component, coherently with the prior knowledge.

The results in Table 3 show the performance obtained on the dataset based on keywords. By inspecting the results obtained across all configurations, it is possible to observe that limiting HURI to the analysis of user relationships yields, in the best case, an F1-Score on all users of 0.726 and an accuracy on borderline users of 0.437. On the other hand, the HURI variant that solely analyzes the textual content achieves in the best configuration a close-to-zero F1-Score performance on all users and an accuracy on borderline users of 0.890. These results are significantly worse than those obtained by HURI analyzing both textual contents and relationships. Indeed, the best results achieved on this dataset using HURI with all active components correspond to an F1-Score of 0.808 on all users and an accuracy of 0.947 on borderline users.

A similar situation is observed with the dataset based on sentiment, as shown in Table 4. Specifically, limiting HURI to the analysis of user relationships yields, in the best case, an F1-Score on all users of 0.389, and an accuracy on borderline users of 0.200. On the other hand, the HURI variant that solely analyzes the textual content achieves in the best configuration an F1-Score performance on all users of 0.172 on and an accuracy on borderline users of 0.840. The full version of HURI largely outperforms such results, achieving an F1-Score on all users of 0.649 and an accuracy on borderline users of 0.949.

Overall, these results confirm that the synergic analysis of both textual content and user relationships, provided by the combination of the semantic and topology components of HURI, is the enabling factor allowing HURI to yield the most accurate predictive performance.

In Tables 1 and 2 we also compare the results of HURI with the results obtained by the two best competitor systems, i.e., MrSBC ($max\_length = 6$) and Doc2Vec+Node2Vec+SVM (kt=128, kc=512) for the keyword dataset, and MrSBC ($max\_length = 6$) and GNetMine for the sentiment dataset. From the results, it is possible to observe that HURI is able to outperform them on both datasets, in almost all the configurations of the parameters k_t and

k_c. The only case where MrSBC outperforms HURI is in on the whole set of users, in the configuration $\langle k_t = 512, k_c = 128 \rangle$ on the Sentiment dataset. However, we can see that it fails to correctly classify borderline users since it classifies all of them as risky.

The best competitors were selected by observing the detailed results obtained by all competitor methods (see Tables 5 and 6). A broader analysis of results obtained by competitor methods reveals that Doc2Vec, with all the considered downstream classifier (Random

Table 3 Ablation study considering simplified variants of HURI (only content and only relationships) on the Keywords dataset

HURI only content					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kc	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.171	0.008	0.015	0.437	0.890
256	0.367	0.009	0.017	0.438	0.860
512	0.333	0.008	0.015	0.437	0.840
HURI only relationships (kt = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.567	0.900	0.723	0.567	0.435
SVD	0.567	0.900	0.723	0.567	0.435
Node2Vec	0.567	0.900	0.723	0.567	0.435
Adjacency Matrix	0.567	0.900	0.723	0.567	0.435
PCA	0.567	0.900	0.723	0.567	0.435
HURI only relationships (kt = 256)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.565	0.900	0.722	0.565	0.433
SVD	0.565	0.900	0.722	0.565	0.433
Node2Vec	0.565	0.900	0.722	0.565	0.433
Adjacency Matrix	0.567	0.900	0.723	0.567	0.435
PCA	0.565	0.900	0.722	0.565	0.433
HURI only relationships (kt = 512)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.569	0.900	0.726	0.569	0.437
SVD	0.569	0.900	0.726	0.569	0.437
Node2Vec	0.569	0.900	0.726	0.569	0.437
Adjacency Matrix	0.567	0.900	0.723	0.567	0.435
PCA	0.569	0.900	0.726	0.569	0.437

Table 4 Ablation study considering simplified variants of HURI (only content and only relationships) on the Sentiment dataset

HURI only content					
kc	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.367	0.128	0.150	0.717	0.820
256	0.395	0.153	0.172	0.710	0.840
512	0.200	0.004	0.008	0.685	0.830
HURI only relationships (kt = 128)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.253	0.800	0.384	0.389	0.200
SVD	0.253	0.800	0.384	0.389	0.200
Node2Vec	0.253	0.800	0.384	0.389	0.200
Adjacency Matrix	0.253	0.800	0.384	0.389	0.200
PCA	0.253	0.800	0.384	0.389	0.200
HURI only relationships (kt = 256)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.265	0.796	0.389	0.403	0.200
SVD	0.265	0.796	0.389	0.403	0.200
Node2Vec	0.265	0.796	0.389	0.403	0.200
Adjacency Matrix	0.253	0.800	0.384	0.389	0.200
PCA	0.265	0.796	0.389	0.403	0.200
HURI only relationships (kt = 512)					
Network representation	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
Autoencoder	0.236	0.763	0.357	0.360	0.200
SVD	0.236	0.763	0.357	0.360	0.200
Node2Vec	0.236	0.763	0.357	0.360	0.200
Adjacency Matrix	0.253	0.800	0.384	0.389	0.200
PCA	0.236	0.763	0.357	0.360	0.200

Forests, SVMs, and Multi Layer Perceptron) was able to obtain acceptable overall results only on the dataset based on keywords. This means that, although it was able to catch the semantics of the content, the learned feature space was not able to properly represent the sentiment of the tweets. We observe an opposite behavior when focusing on the accuracy of the classification of *borderline* users, that indicates that this approach was not able to properly handle the noise in the data. In other words, it was not possible to simultaneously achieve an acceptable overall F1-Score and a good accuracy on *borderline* users.

The system *GNetMine*, which exclusively analyzes the link structure of the network, generally exhibited a poor classification performance on the whole dataset, but a high accuracy on possible *borderline* users. However, a close analysis of the prediction results reveals that

Table 5 Average performance for all competitor methods on the Keywords dataset

GNetMine					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
	0.580	0.464	**0.515**	0.564	**0.657**
Doc2Vec + Random Forest (α = 0.0)					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.493	0.483	0.488	0.493	0.491
256	0.505	0.487	**0.496**	0.506	**0.521**
512	0.479	0.480	0.479	0.478	0.498
Doc2Vec + Node2Vec + Random Forest (α = 0.0)					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.661	0.648	**0.601**	0.641	0.341
kt = 128, kc = 512	0.577	0.602	0.588	0.641	0.393
kt = 256, kc = 128	0.529	0.657	0.587	0.629	**0.405**
kt = 512, kc = 128	0.534	0.682	0.600	0.621	0.280
Doc2Vec + Support Vector Machine					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.498	0.490	0.494	0.498	**0.543**
256	0.506	0.516	**0.511**	0.506	0.475
512	0.493	0.484	0.488	0.493	0.517
Doc2Vec + Node2Vec + Support Vector Machine					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.724	0.377	0.495	0.567	0.667
kt = 128, kc = 512	0.729	0.398	0.513	0.575	**0.684**
kt = 256, kc = 128	0.733	0.363	0.484	0.565	0.682
kt = 512, kc = 128	0.695	0.472	**0.562**	0.585	0.515
Doc2Vec + Multi Layer Perceptron					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.643	0.631	0.636	0.592	0.386
256	0.669	0.650	**0.659**	0.621	**0.455**
512	0.657	0.647	0.651	0.609	0.446

Table 5 continued

GNetMine					
Doc2Vec + Node2Vec + Multi Layer Perceptron					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.640	0.541	**0.585**	0.567	0.517
kt = 128, kc = 512	0.625	0.528	0.572	0.553	0.509
kt = 256, kc = 128	0.597	0.502	0.545	0.527	**0.535**
kt = 512, kc = 128	0.609	0.541	0.572	0.544	0.487
MrSBC					
Max Length Path	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
3	0.250	0.500	0.334	0.250	0.000
4	0.250	0.500	0.334	0.250	0.000
5	0.250	0.500	0.334	0.250	0.000
6	0.500	1.000	**0.667**	0.500	0.000
Best Competitors					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
MrSBC (*max_length*=6)	0.500	1.000	**0.667**	0.500	0.000
d2v+n2v+SVM (kt=128, kc=512)	0.729	0.398	0.513	0.575	**0.684**

The best results for each configuration in terms of F1-Score and accuracy of classification of borderline users are highlighted in bold

the method is very prone to classify users as *safe*, that is, the topology alone does not appear sufficient to accurately detect high-risk users. Consequently, the positive results on *borderline* users do not imply an actual discriminating ability.

The hybrid method *MrSBC* shows a weak classification performance on the whole dataset and a relatively small accuracy on *borderline* users in all its parameter configurations. Shifting our focus on the hybrid method *Doc2Vec + Node2Vec* (with all the considered downstream classifiers), it is possible to observe significantly more accurate results than *MrSBC* in terms of accuracy on *borderline* users, but worse results in terms of F1-Score, on both datasets. The highest accuracy on *borderline* users is achieved with Random Forest on the dataset based on sentiment, and Support Vector Machine on the dataset based on keywords. The situation is reversed when observing results for *Doc2Vec + Node2Vec* in terms of F1-Score on all users, i.e. Support Vector Machine is the best classifier on the dataset based on sentiment, whereas Random Forest is the leading classifier on the dataset based on keywords. This behavior shows that the adoption of a method able to catch both the content and the relationships does not necessarily guarantee an accurate classification, especially if the method does not explicitly exploit the semantics associated to the content (which is the case of *MrSBC*).

In conclusion, the proposed method HURI showed the best overall performance, in both discriminating between *safe* and *risky* users and in being robust to the presence of noisy data, i.e., *borderline* users represented by journalists. This superiority is observable with both considered datasets, meaning that HURI was able to correctly leverage the semantics from

Table 6 Average performance for all competitor methods on the Sentiment dataset

GNetMine					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
	0.388	0.195	**0.247**	0.694	**0.898**
Doc2Vec + Random Forest (α = 0.2)					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.062	0.200	0.094	0.615	**0.800**
256	0.124	0.400	0.189	0.538	0.600
512	0.185	0.600	**0.283**	0.462	0.400
Doc2Vec + Random Forest (α = 0.2)					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.454	0.142	0.215	0.674	0.743
kt = 128, kc = 512	0.466	0.145	**0.221**	0.678	0.763
kt = 256, kc = 128	0.438	0.120	0.187	0.672	**0.770**
kt = 512, kc = 128	0.441	0.113	0.179	0.673	0.760
Doc2Vec + Support Vector Machine					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.327	0.346	**0.335**	0.578	0.651
256	0.301	0.286	0.293	0.575	0.693
512	0.328	0.290	0.308	0.597	**0.715**
Doc2Vec + Node2Vec + Support Vector Machine					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.363	0.384	0.372	0.593	0.610
kt = 128, kc = 512	0.358	0.382	0.369	0.589	**0.613**
kt = 256, kc = 128	0.365	0.398	**0.377**	0.592	0.597
kt = 512, kc = 128	0.364	0.389	0.370	0.592	0.590
Doc2Vec + Multi Layer Perceptron					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
128	0.429	0.207	0.279	0.662	**0.809**
256	0.410	0.203	**0.271**	0.655	0.776
512	0.427	0.158	0.225	0.667	0.806

Table 6 continued

GNetMine					
Doc2Vec + Node2Vec + Multi Layer Perceptron					
Vector Dimensionality	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
kt = 128, kc = 256	0.338	0.296	0.315	0.593	0.677
kt = 128, kc = 512	0.360	0.337	**0.347**	0.600	0.669
kt = 256, kc = 128	0.355	0.320	0.336	0.602	**0.683**
kt = 512, kc = 128	0.356	0.332	0.342	0.599	0.660
MrSBC					
Max Length Path	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
3	0.155	0.500	0.236	0.156	0.000
4	0.155	0.500	0.236	0.156	0.000
5	0.155	0.500	0.236	0.156	0.000
6	0.310	1.000	**0.473**	0.313	0.000
Best Competitors					
	All Users				Borderline Users
	Precision	*Recall*	*F1-Score*	*Accuracy*	*Accuracy*
MrSBC (max_length=6)	0.310	1.000	**0.473**	0.313	0.000
GNetMine	0.388	0.195	0.247	0.694	**0.898**

The best results for each configuration in terms of F1-Score and accuracy of classification of borderline users are highlighted in bold

the content in both situations and to properly combine its contribution with of the network structure.

Finally, in Figure 5 we show an example of words appearing in Twitter posts that HURI classified as *safe* (Figure 5.a) and *risky* (Figure 5.b), respectively, according to the semantic content analysis of the system. By observing the figure, it is clear that the semantic component of our method can accurately classify users. Nevertheless, as previously emphasized, the final user classification still depends on the combined contribution of the content-based and topology-based components which allow us to accurately label *borderline* users as *safe* users.

6 Conclusion

In this paper, we have proposed HURI, a method for social network analysis that exploits multiple sources of information to accurately classify users as safe or risky. In our method, we have simultaneously leveraged the network topology and the semantics of the content shared by users to analyze in detail the underlying social relationships and interactions. This was possible thanks to the stacked generalization approach proposed in HURI, which learns an adaptive model to combine the two contributions.

The experimental results showed that the proposed method exhibits competitive performance with respect to topology-based, content-based, and hybrid state-of-the-art approaches

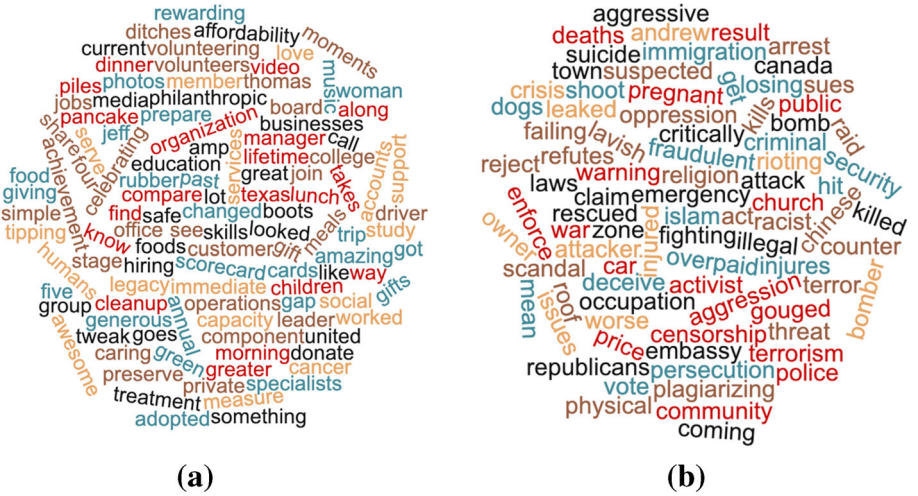

Fig. 5 An example of words appearing in Twitter posts that HURI classifies as *safe* (a) or *risky* (b) according to the semantic analysis component. The final user classification in any case depends on the combined contribution of the content-based and topology-based components

for social network analysis, especially in the presence of noisy data. We analyzed the performance of the different methods in a complex network scenario that includes *borderline* users who, in this specific context, may represent journalists who post contents that may appear risky, but who are actually *safe* users according to their relationships.

We observed that all the competitor methods analyzed provide unsatisfactory performances either in terms of classification accuracy on all the users or specifically on *borderline* users. On the contrary, our method provides the best results on both the considered tasks. One possible limitation of the proposed method HURI is a potential reduction of accuracy in scenarios characterized by *borderline* users with an unknown or ill-defined network topology. Another potential challenge for the method is the increased difficulty for the classification task arising when *borderline* users mimic both the topology and the generated content of risky users. In the future, we aim to assess and improve the robustness of HURI in such situations, and to extend it to address complex applications that involve multi-modal data, including images and videos. Moreover, we will design a distributed variant of HURI able to analyze large-scale networks.

Acknowledgements We acknowledge the support of Advanced Symbolics Inc. for providing us with the dataset used in our study. We also acknowledge the support of NVIDIA through the donation of a Titan V GPU. Finally, we would like to thank Lynn Rudd for her help in reading the manuscript.

Funding The authors acknowledge the support of the European Commission through the H2020 Projects "CounteR - Privacy-First Situational Awareness Platform for Violent Terrorism and Crime Prediction, Counter Radicalisation and Citizen Protection" (Grant no. 101021607) and "IMPETUS - Intelligent Management of Processes, Ethics and Technology for Urban Safety" (Grant. no. 883286). We also acknowledge the support of the U.S. Defense Advanced Research Projects Agency (DARPA) through the project "Lifelong Streaming Anomaly Detection" (Grant no. A19-0131-003 and A21-0113-002). Dr. Gianvito Pio acknowledges the support of Ministry of Universities and Research (MUR) through the project "Big Data Analytics", AIM 1852414, activity 1, line 1. This work was also partially supported by the project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI, under the NRRP MUR program funded by the NextGenerationEU.

Availability of data and materials The system HURI and the results obtained with all the configurations are available at http://www.di.uniba.it/~gianvitopio/systems/huri/.

Declarations

Conflict of interest/Competing interests The authors have no relevant financial or non-financial interests to disclose.

References

1. Huang, B., Raisi, E.: Weak Supervision and Machine Learning for Online Harassment Detection, Springer, Cham pp 5–28 (2018)
2. Awan, I.: Cyber-Extremism: Isis and the Power of Social Media. Society **54**(2), 138–149 (2017)
3. Al-Rawi, A., Groshek, J.: Jihadist Propaganda on Social Media: An Examination of ISIS Related Content on Twitter. Int J Cyber Warfare and Terrorism (IJCWT) **8**(4), 1–15 (2018)
4. Alfifi, M., Kaghazgaran, P., Caverlee, J., Morstatter, F.: 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**, 58–67 (2019)
5. Shaheen, J., et al.: Network of Terror: How Daesh Uses Adaptive Social Networks To Spread Its Message. NATO Strategic Communications Centre of Excellence, Riga, Latvia (2015)
6. Ji, M., Sun, Y., Danilevsky, M., Han, J., Gao, J.: Graph regularized transductive classification on heterogeneous information networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer pp. 570–586 (2010)
7. Macskassy, S.A., Provost, F.: Classification in networked data: A toolkit and a univariate case study. Journal of Machine Learning Research 935–983 (2007) 8 May
8. Gallagher, B., Tong, H., Eliassi-Rad, T., Faloutsos, C.: Using ghost edges for classification in sparsely labeled networks. In: Proc. of SIGKDD Int. Conference on Knowledge Discovery and Data Mining, ACM pp. 256–264 (2008)
9. Bilgic, M., Getoor, L.: Effective label acquisition for collective classification. In: Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '08, ACM, New York pp. 43–51 (2008)
10. Zhou, W., Han, C., Huang, X.: Multiclass classification of tweets and twitter users based on kindness analysis. In: CS229 Final Project Report (2016)
11. Uzel, V.N., Saraç Eşsiz, E., Ayşe Özel, S.: Using fuzzy sets for detecting cyber terrorism and extremism in the text. In: 2018 Innovations in Intelligent Systems and Applications Conference (ASYU) pp. 1–4 (2018)
12. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International Conference on Machine Learning pp. 1188–1196 (2014)
13. Pio, G., Serafino, F., Malerba, D., Ceci, M.: Multi-type clustering and classification from heterogeneous networks. Inf. Sci. **425**, 107–126 (2018)
14. Ceci, M., Appice, A., Malerba, D.: Mr-SBC: A Multi-relational Naïve Bayes Classifier. In: Proc. of Knowledge Discovery in Databases: PKDD 2003 pp. 95–106 (2003)
15. Serafino, F., Pio, G., Ceci, M.: Ensemble learning for multi-type classification in heterogeneous networks. IEEE Trans. Knowl. Data Eng. **30**(12), 2326–2339 (2018)
16. Campbell, W., Baseman, E., Greenfield, K.: Content+context networks for user classification in twitter. In: Frontiers of Network Analysis: Methods, Models, and Applications Workshop at Neural Information Processing Systems (2013)

17. Xie, D., Xu, J., Lu, T.: Automated classification of extremist twitter accounts using content-based and network-based features. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 2545–2549 (2016)
18. Bengio, Y., et al: Learning deep architectures for AI. Foundations and trends® in Machine Learning **2**(1), 1–127 (2009)
19. Wolpert, D.H.: Stacked generalization. Neural Networks **5**(2), 241–259 (1992)
20. Otte, E., Rousseau, R.: Social network analysis: a powerful strategy, also for the information sciences. Journal of Information Science **28**(6), 441–453 (2002)
21. Camacho, D., Panizo-LLedot, Á., Bello-Orgaz, G., Gonzalez-Pardo, A., Cambria, E.: The four dimensions of social network analysis: An overview of research methods, applications, and software tools. Inf. Fusion **63**, 88–120 (2020)
22. Scott, J.: Social network analysis. Sociology **22**(1), 109–127 (1988)
23. Bartal, A., Sasson, E., Ravid, G.: Predicting Links in Social Networks Using Text Mining and SNA. In: 2009 International Conference on Advances in Social Network Analysis and Mining pp. 131–136 (2009)
24. Sadayappan, S., McCulloh, I., Piorkowski, J.: Evaluation of political party cohesion using exponential random graph modeling. In: IEEE/ACM ASONAM 2018 pp. 298–301 (2018)
25. Karimi, H., VanDam, C., Ye, L., Tang, J.: End-to-end compromised account detection. In: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM) pp. 314–321 (2018)
26. Crandell, I., Korkmaz, G.:Link prediction in the criminal network of albuquerque.In: IEEE/ACM ASONAM 2018 pp. 564–567 (2018)
27. Choudhary, P.: A survey on social network analysis for counter-terrorism. Int J Comput Appl **112** (2015)
28. Gialampoukidis, I., Kalpakis, G., Tsikrika, T., Vrochidis, S., Kompatsiaris, I.: Key player identification in terrorism-related social media networks using centrality measures. In: EISIC 2016, pp. 112–115 (2016)
29. Farooq, E., Khan, S.A., Butt, W.H.: Covert network analysis to detect key players using correlation and social network analysis. In: Proc. of the Second International Conference on Internet of Things, Data and Cloud Computing. ICC '17, ACM,New York pp. 94–1946 (2017)
30. Gialampoukidis, I., Kalpakis, G., Tsikrika, T., Papadopoulos, S., Vrochidis, S., Kompatsiaris, I.:Detection of terrorism-related twitter communities using centrality scores. In: Proc. of the 2Nd Int. Workshop on Multimedia Forensics and Security. MFSec '17, ACM,New York pp. 21–25 (2017)
31. Saidi, F.,Trabelsi, Z.,Ghazela, H.B.: A novel approach for terrorist sub-communities detection based on constrained evidential clustering. In: Proc. of Int. Conf. on Res. Challenges in Information Science, pp. 1–8 (2018)
32. Wiil, U.K., Gniadek, J., Memon, N.: Measuring link importance in terrorist networks. In: 2010 International Conference on Advances in Social Networks Analysis and Mining pp. 225–232 (2010)
33. Zhou, Y., Reid, E., Qin, J., Chen, H., Lai, G.: US domestic extremist groups on the Web: link and content analysis. IEEE Intell. Syst. **20**(5), 44–51 (2005)
34. Kaza, S., Hu, D., Chen, H.: Dynamic social network analysis of a dark network: Identifying significant facilitators. In: 2007 IEEE Intelligence and Security Informatics pp. 40–46 (2007)
35. Adler, R.M.: A dynamic social network software platform for counter-terrorism decision support. In: IEEE ITSS 2007 pp. 47–54 (2007)
36. Wang, Y., Zhu, L.: Research and implementation of svd in machine learning. In: 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS) pp. 471–475 (2017)
37. Jolliffe, I., Cadima, J.: Principal component analysis: A review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **374**, 20150202 (2016)
38. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature **401**, 788–791 (1999)
39. Buono, N.D., Pio, G.: Non-negative matrix tri-factorization for co-clustering: An analysis of the block matrix. Inf. Sci. **301**, 13–26 (2015)
40. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proc. of SIGKDD Int. Conference on Knowledge Discovery and Data Mining. KDD '14, ACM,New York pp. 701–710 (2014)
41. Grover, A., Leskovec, J.: Node2vec: Scalable feature learning for networks. In: Proc. of SIGKDD Int. Conference on Knowledge Discovery and Data Mining. KDD '16, ACM, New York, NY, USA pp. 855–864 (2016)
42. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proc. of Int. Conference on World Wide Web pp. 1067–1077 (2015)

43. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM pp. 1225–1234 (2016)
44. Liu, J., He, Z.,Huang, Y.: Hashtag2Vec: Learning Hashtag Representation with Relational Hierarchical Embedding Model. In: Proc. of IJCAI 2018 pp. 3456–3462 (2018)
45. Du, Y., Guo, W., Liu, J., Yao, C.: Classification by multi-semantic meta path and active weight learning in heterogeneous information networks. Expert Systems with Applications **123**, 227–236 (2019)
46. Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: Proc. of SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM pp. 593–598 (2004)
47. Jethava, G., Rao, U.P.: User behavior-based and graph-based hybrid approach for detection of sybil attack in online social networks. Computers and Electrical Engineering **99**, 107753 (2022)
48. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems pp. 3111–3119 (2013)
49. Lara-Cabrera, R., Gonzalez-Pardo, A., Camacho, D.: Statistical analysis of risk assessment factors and metrics to evaluate radicalisation in Twitter. Future Generation Computer Systems **93**, 971–978 (2019)
50. Abbasi, F., Fazl-Ersi, E.: Identifying influentials in social networks. Applied Artificial Intelligence **36**(1), 2010886 (2022)
51. Bhih, A., Johnson, P., Randles, M.: An optimisation tool for robust community detection algorithms using content and topology information. J Supercomput **76**(1), 226–254 (2020)
52. Martinez-Romo, J., Araujo, L.: Detecting malicious tweets in trending topics using a statistical analysis of language. Expert Systems with Applications **40**(8), 2992–3000 (2013)
53. Desrosiers, C., Karypis, G.: Within-network classification using local structure similarity. In: ECML PKDD '09 pp. 260–275 (2009)
54. Barracchia, E.P., Pio, G., Bifet, A., Gomes, H.M., Pfahringer, B., Ceci, M.: LP-ROBIN: Link prediction in dynamic networks exploiting incremental node embedding. Information Sciences **606**, 702–721 (2022)
55. Lu, Q.,Getoor, L.: Link-based classification using labeled and unlabeled data. In: ICML Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining (2003)
56. Stojanova, D., Ceci, M., Appice, A., Dzeroski, S.: Network regression with predictive clustering tree. Data Min. Knowl. Discov. **25**(2), 378–413 (2012)
57. Hinton, G., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. science **313**(5786), 504–507 (2006)
58. Cai, H., Zheng, V.W., Chang, K.C.: A comprehensive survey of graph embedding: Problems, techniques, and applications. IEEE Transactions on Knowledge and Data Engineering **30**(9), 1616–1637 (2018)
59. Levatic, J., Kocev, D., Ceci, M., Dzeroski, S.: Semi-supervised trees for multi-target regression. Inf. Sci. **450**, 109–127 (2018)
60. Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. CRC press, ??? (1984)
61. Mirończuk, M.M., Protasiewicz, J.: A recent overview of the state-of-the-art elements of text classification. Expert Systems with Applications **106**, 36–54 (2018)
62. Japkowicz, N.: Supervised versus unsupervised binary-learning by feedforward neural networks. Machine Learning **42**(1/2), 97–122 (2001)
63. Corizzo, R., Ceci, M., Japkowicz, N.: Anomaly detection and repair for accurate predictions in geo-distributed big data. Big Data Res. **16**, 18–35 (2019)
64. Corizzo, R., Ceci, M., Zdravevski, E., Japkowicz, N.: Scalable auto-encoders for gravitational waves detection from time series data. Expert Systems with Applications **151**, 113378 (2020)
65. Bellinger, C., Sharma, S., Japkowicz, N.: One-class versus binary classification: Which and when? In: 2012 11th International Conference on Machine Learning and Applications **2**, pp. 102–106 (2012)
66. Haykin, S.: Neural Networks: a Comprehensive Foundation. Prentice Hall PTR, New Jersey, United States (1994)
67. Karlik, B., Olgac, A.V.: Performance analysis of various activation functions in generalized mlp architectures of neural networks. Int J Artif Intell Expert Syst **1**(4), 111–122 (2011)
68. Sheela, K.G., Deepa, S.N.: Review on methods to fix number of hidden neurons in neural networks. Mathematical Problems in Engineering **2013** (2013)
69. Garavaglia, S., Sharma, A.: A smart guide to dummy variables: Four applications and a macro. In: Proc. of the Northeast SAS Users Group Conference p. 43 (1998)
70. White, K., Li, G., Japkowicz, N.: Sampling online social networks using coupling from the past. In: Proc. of IEEE International Conference on Data Mining Workshops pp. 266–272 (2012)

71. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit. Proc. of Annual Meeting of the Association for Computational Linguistics: System Demonstrations 55–60 (2014)
72. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures. In: Neural Networks: Tricks of the Trade, Springer,Berlin pp. 437–478 (2012)
73. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., et al.: Scikit-learn: Machine learning in Python. J Mach Learning Research **12**, 2825–2830 (2011)

Authors and Affiliations

Roberto Corizzo[1] · Gianvito Pio[2,3] · Emanuele Pio Barracchia[2,3] · Antonio Pellicani[2,3] · Nathalie Japkowicz[1] · Michelangelo Ceci[2,3,4]

Roberto Corizzo
rcorizzo@american.edu

Gianvito Pio
gianvito.pio@uniba.it

Emanuele Pio Barracchia
emanuele.barracchia@uniba.it

Antonio Pellicani
antonio.pellicani@uniba.it

Nathalie Japkowicz
japkowic@american.edu

1 Department of Computer Science, American University, 4400 Massachusetts Ave NW, Washington 20016, DC, United States

2 Department of Computer Science, University of Bari Aldo Moro, Via E. Orabona, 4, Bari 70125, Italy

3 Big Data Laboratory, National Interuniversity Consortium for Informatics (CINI), Via Volturno, 58, Roma 00185, Italy

4 Department of Knowledge Technologies, Jožef Stefan Institute, Jamova 39, Ljubljana 1000, Slovenia