ContCommRTD: A Distributed Content-Based Misinformation-Aware Community Detection System for Real-Time Disaster Reporting

Elena-Simona Apostol, Ciprian-Octavian Truică, and Adrian Paschke

***Abstract*—Real-time social media data can provide useful information on evolving hazards. Alongside traditional methods of disaster detection, the integration of social media data can considerably enhance disaster management. In this paper, we investigate the problem of detecting geolocation-content communities on Twitter and propose a novel distributed system that provides in near real-time information on hazard-related events and their evolution. We show that content-based community analysis can lead to better and faster dissemination of hazard-related reports than using only traditional methods, such as satellite or airborne sensing platforms. Our distributed disaster reporting system analyzes the social relationship among worldwide geolocated tweets and applies topic modeling to group tweets by topics. Considering for each tweet the following information: user, timestamp, geolocation, retweets, and replies, we create a publisher-subscriber distribution model for topics. We use content similarity and the proximity of nodes to create a new model for geolocation-content based communities. Users can subscribe to different topics in specific geographical areas or worldwide and receive real-time reports regarding these topics. As misinformation can lead to increased damage if propagated in hazards-related tweets, we propose a new deep learning model to detect fake news. The misinformed tweets are then removed from display. We also show empirically the scalability capabilities of the proposed system.**

***Index Terms*—Community detection, distributed system, misinformation detection, spatial-temporal system, topic modeling.**

Manuscript received 9 June 2022; revised 18 December 2023; accepted 17 June 2024. Date of publication 24 June 2024; date of current version 27 September 2024. This work was supported in part by the National Program for Research of the National Association of Technical Universities under Grant GNAC ARUT 2023) through the project "DEPLATFORM: Intelligent interactive system for detecting the veracity of news published on social platforms" (Contract no. 63/10.10.2023), in part by the German Academic Exchange Service (DAAD) through the project "iTracing: Automatic Misinformation Fact-Checking" DAAD under Grant 91809005, in part by the German Federal Ministry of Education and Research (BMBF) project "PANQURA - a technology platform for more information transparency in times of crisis" under Grant 03COV03F, and in part by the European Union project "FAST-LISA - Fighting hAte Speech Through a Legal, ICT and Sociolinguistic approach" under Grant 101049342. Recommended for acceptance by A. M. K. Cheng. *(Elena-Simona Apostol and Ciprian-Octavian TRUICĂ contributed equally to this work.) (Corresponding author: Ciprian-Octavian Truică.)*

Ciprian-Octavian Truică is with the Computer Science and Engineering Department, Faculty of Automatic Control and Computers, National University of Science and Technology Politehnica Bucharest, 060042 Bucharest, Romania (e-mail: ciprian.truica@upb.ro).

Elena-Simona Apostol is with the Computer Science and Engineering Department, Faculty of Automatic Control and Computers, National University of Science and Technology Politehnica Bucharest, 060042 Bucharest, Romania, and also with the Academy of Romanian Scientists, 3 Ilfov, 050044 Bucharest, Romania (e-mail: elena.apostol@upb.ro).

Adrian Paschke is with the Fraunhofer Institute for Open Communication Systems, 10589 Berlin, Germany (e-mail: adrian.paschke@fokus.fraunhofer.de).

Digital Object Identifier 10.1109/TKDE.2024.3417232

I. Introduction

Worldwide, hazards frequently have a dramatic impact on rural and urban societies or the environment. There are different classes of hazards, e.g., natural (geophysical, hydrological, etc.), anthropogenic (chemical, major accident, technological, biological, etc.). In recent years many studies have recognized the value of community activity during disasters due to hazards [1], [2]. Following these analyses, it was observed that people who are physically near a location where a hazard occurred tend to produce more disaster-related information on social media [3], [4].

Alongside traditional methods of hazard detection, e.g., satellite or airborne sensing platforms, topographic data sources, Internet of Things, the integration of social media data would considerably enhance disaster management, especially in areas with little to no infrastructure, and, therefore, will be of greater importance in the future [5], [6]. Twitter is one of the largest social networking platforms, and it has a remarkable capability of continuous retrieval of data and content-sharing services [7]. It is also a suitable place where citizens can present their concerns in a real-time manner [8].

Social media has a significant impact on public perception of different hazards and their contingency plans. This is why is extremely important not to consider tweets that contain misinformation. Although several papers examined the negative impact of misinformation spread on Twitter during disasters that can cause mass panic or financial loss [9], in the current literature there is no work that mitigates their harm in real-time.

In this study, we attempt to answer a fundamental question: Can the social and geographical disparities of Twitter be used to assist citizens and government organizations in informing or disaster management? To answer this question, we consider classifying tweets into different disaster-related topics and constructing a graph model based on content and geolocation. We also recognize the ethical issues caused by the fact that tweets collected from unreliable sources and shown to other users may increase the spread of misinformation [10]. Thus, we also apply fake news detection before displaying the data to the end user.

In this paper, we 1) investigate the problem of detecting content-based communities in order to provide valuable information on hazard-related events and their evolution, and

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

2) propose a distributed content-based misinformation-aware community detection system for real-time disaster reporting (ContCommRTD). Besides the proposed system, the main contributions are:

1) We propose a new distributed solution to analyze the social relationship among worldwide geolocated tweets.
2) We define a new social network model for detecting geolocation-content based communities and named entities.
3) We create a meta-knowledge dictionary to collect topic-specific tweets containing hazard-related keywords and hashtags.
4) We apply topic modeling to discover topics and to classify tweets into different topics. A user can then subscribe to one of these topics and receive in real-time all the tweets that discuss the event of interest.
5) We propose an efficient distributed publish-subscribe based model using Apache Kafka [11], [12] and MongoDB [13] that allows real-time collection, storage, and distribution of Twitter posts relevant to the chosen type of hazard.
6) We propose a new Deep Learning architecture, FN-BERT-TFIDF, to determine if the tweets of interest are fake news.

The rest of this paper is structured as follows: In Section II, we present the state-of-the-art. In Section IV, we describe the proposed system. In Section V, we present the datasets and analyze our results. Section VI concludes the paper and hints at future research.

II. Related Work

Real-time disaster detection using data from dynamic social media environments, e.g., Twitter, and considering different locations is a challenging task that recently received much attention [14], [15], [16]. Ferner et al. (2020) [17] propose detecting disaster-related topics on Twitter data using an LDA (Latent Dirichlet Allocation) topic model enhanced with a set of seed words from older Tweets of the same geographic area. This solution is applicable when there is a single topic of interest, e.g., earthquakes or hurricanes, and similar data is available. Several solutions use in the detection task not only the textual data from tweets but also incorporate images from the posts or satellites [18], [19], [20]. Arachie et al. (2020) [21] present an unsupervised learning solution that detects large-scale hazard-related sub-events in Tweets. The authors use an ontology containing crisis management vocabulary to rank the candidate sub-events and then cluster the most important sub-events using spectral clustering.

Geolocation is also important when considering natural hazards analysis using social media data. To the best of our knowledge, the current real-time unsupervised disaster detection solutions consider only geotagged tweets [22], [23]. However, tweets do not necessarily come with geoinformation. Several offline classifier-based disaster detection solutions also try to determine the location by searching the tweets' content for names of cities and countries [24], [25]. These solutions use corpora of labeled tweets. Loynes et al. (2020) [24] propose using GeoText to search for names of cities and countries. Another solution proposes location mention recognition from labeled crisis-related tweets using BERT (Bidirectional Encoder Representations from Transformers [26]) based classification [25].

In the current literature, different graph-based methods are applied to citizen science-based disaster detection solutions. Dou et al. (2021) [27] use a semantic graph-based topic detection method to identify fine-grained topics during natural disasters in social media. A community detection algorithm, i.e., the algorithm of modularity optimization [28], is also used to extract topics that denote the same class of event information. A similar pipeline is also proposed in article [29]: 1) a graph generation algorithm is used to transform the text data into a graph of the keywords, 2) a community detection algorithm is applied to discover hazard-related topics. Nguyen et al. (2021) [30] present a framework for 1) filtering and classifying tweets and 2) identifying and summarizing important disaster-related topics. The authors use a graph-based ranking algorithm to select and summarize important tweets.

Another issue to consider regarding this research topic is the impact of misinformation. Misinformation (false or inaccurate information) and disinformation (intentionally spreading misinformation) in the form of fake news are tools used to manipulate public opinion on particular topics, distort public perceptions, and generate social unrest while lacking the rigor of traditional journalism [10], [31], [32], [33], [34], [35], [36]. Singh et al. (2020) [37] propose content and context-aware RNN-based solution for fake news detection during natural disasters. Their solution uses the user profile information and the temporal and textual features of the analyzed events. Pelrine et al. (2021) [38] analyze the performance for misinformation detection of several transformer language models (e.g., BERT [26], RoBERTa [39], ALBERT [40]) on different datasets. Based on their results, these transformer models have very good performance metrics for large enough datasets. However, if we consider smaller datasets, such as the case of disaster-related Twitter datasets (e.g., COAID [41]), the performance decreases. Although several studies have analyzed the impact of misinformation on different types of hazards, e.g., Hurricane [42], COVID-19 [43], none offer a real-time hazard-related event detection solution while removing misinformation. CNN and CNN-RNN models have provided good results when trained on Twitter data to tackle misinformation [32], [33]. Transformer-based models [10], [34] also show promising results for misinformation detection. The most important feature to consider when dealing with this task is the way documents are vectorized [35]. Most of the time, a simpler neural architecture produces better or at least comparable outcomes to complex architectures. The difference in performance leans in the employed embeddings.

As far as we know, there are not many holistic event-based systems that analyze and detect topics of interest on social media and in real-time during various natural disasters. TriggerCit [44] is an early flood alerting tool that monitors social media content (i.e., tweets) and focuses on detecting the occurrence of flood-related messages. This solution uses a seed dictionary for floods and a simplistic approach for detecting flood-related tweets (i.e., word count).

To summarize, the main shortcomings of the solutions analyzed by us are, as follows. 1) The majority of current citizen science-based disaster analyzing solutions focus only on the task of detecting disaster-related topics. 2) Although several studies have analyzed the impact of misinformation on different types of hazards, these solutions only focus on misinformation detection and not on detecting disaster-related topics or triggering alerts based on the analyzed content.

III. Problem Definition and Methodology

We define an undirected graph $G = (V, E, C, L)$, where V is the set of vertices, E is the set of edges, C is the set that stores the textual content of each vertex, and L the set of geolocations for each node. Thus, for our problem, we define the social network as n undirected graphs $\Gamma = \{G_i = (V_i, E_i, C_i, L_i) | u, v \in V_i \wedge (u, v) \in E_i \wedge c_u, c_v \in C_i \wedge l_u, l_u \in L_i \wedge i = \overline{1, N}\}$, where every two vertices $u, v \in V_i$ are linked by through an edge $(u, v) \in E_i$ that represents the social relationship between them. Each vertex u, v models a tweet, where for each tweet, we also store additional information, e.g., content, geo-location, etc., while edges represent social interactions, e.g., retweets, replies, likes, etc. Thus, Γ is a graph with n disconnected components G_i. The content c_u, c_v of vertices u, v is stored in C_i, while their geolocation l_u, l_v are stored in L_i. We define a social relationship as the link that occurs between these nodes in either of the cases:

1) u retweets the content of v,
2) u replies to v.

The overall approach for detecting content-based communities to provide valuable information on hazard-related events and their evolution is:

1) Hazard-relevant tweets are extracted using a meta-dictionary;
2) A Social Graph is created based on the retweet and reply relationship;
4) Tweets that spread misinformation are removed from the Social Graph;
5) Topic graphs are extracted using a topic modeling algorithm to group tweets that talk about the same hazards together;
6) Geolocation-based community detection is used to group by location tweets that talk about the same hazard;

For each hazard-related event, we have a Meta-Knowledge dictionary that contains a list H with the most relevant keywords and hashtags. The dictionary is constructed separately. Using H, we collect the data from the social network. For each record u, we verify if the location l_u is given. If l_u is missing, we use Name Entity Recognition (NER) to extract any mention of locations and then map the name with its coordinates (geolocation). Using this information together with the social relationships, we construct the list of n undirected graphs Γ, i.e., the social graph.

Algorithm 1 presents the collection and creation of the social graph Γ, which receives as input the list of relevant keywords and hashtags for an event H. Lines 1 to 2 initializes the social graph Γ and the disconnected graph Ω used for extracting the connected components G_i. Lines 3 to 6 initializes the components of Ω The records are collected into R using a Social Network API (Line 7). For each record $u \in R$, we add elements to the V, C, L components of Ω (Lines 8 to 17). If the tweet is not

Algorithm 1: $SocialGraph$ - Construct the Social Graph.

Input : the keywords and hashtags list H
Output: the undirected social graph Γ
1 $\Gamma \leftarrow \emptyset$
2 $\Omega \leftarrow \emptyset$
3 $V \leftarrow \emptyset$
4 $E \leftarrow \emptyset$
5 $C \leftarrow \emptyset$
6 $L \leftarrow \emptyset$
7 $R \leftarrow SocialNetworkAPI(H)$
8 **foreach** $u \in R$ **do**
9   **if** $l_u = Nil$ **then**
10         $\lambda \leftarrow NER(c_u)$
11   **if** $\lambda \neq Nil$ **then**
12             $l_u \leftarrow GeoLocation(\lambda)$
13   **if** $l_u \neq Nil$ **then**
14         $V \leftarrow V \cup \{u\}$
15         $c'_u \leftarrow Preprocess(c_u)$
16         $C \leftarrow C \cup \{c'_u\}$
17         $L \leftarrow L \cup \{l_u\}$
18 **foreach** $u \in V$ **do**
19   **foreach** $v \in V$ **do**
20   **if** $Retweet(u, v) \vee Reply(u, v)$ **then**
21             $E \leftarrow E \cup \{(u, v)\}$
22 $\Omega = (V, E, C, L)$
23 $\Gamma \leftarrow ConnectedComponents(\Omega)$
24 **return** Γ

geo-tagged and the geo-location cannot be extracted using NER, then the tweet is discarded. Line 15 applies some preprocessing techniques on the textual content. Lines 9 to 11 determine the coordinates based on locations if the geolocation is not provided in the record. The edges are added separately after we have all the nodes (Lines 18 to 21). We construct Ω (Line 22), extract the connected components to build Γ (Line 23), and return the social graph (Line 24).

Using the graph structures in Γ, we want to determine communities where there is a high social media activity for users' topics of interest. These communities can contain multiple graphs $G_i \in \Gamma$ that are not directly interconnected. To build the communities, first, we filter the content of a graph G_i and remove any nodes and their edges that spread misinformation. At the end of the misinformation detection process, we obtain Γ' that contains only the clean undirected graphs G'_i.

Algorithm 2 presents the misinformation detection algorithm, which receives as input the social graphs Γ and outputs the clean graph Γ'. After the clean graph is initialized Γ' (Line 1), the veracity of all the nodes' content from a graph $G \in \Gamma$ is verified (Lines 2 to 12). If the content is deemed as Fake, the record is removed from the graph (Lines 4 to 10. The corresponding graph G'_i is constructed using the updated V', E', C', L' and it is added to Γ' (Lines 11 to 12). When the verification is finished for all the graphs, the algorithm returns the clean graph Γ'.

With the clean content Γ', we build the content-based communities. In order to achieve this, we utilize:

Algorithm 2: $MisinformationDetection$ - Content Veracity Detection.

```
Input  : the social graphs Γ
Output: the clean undirected graph Γ'
Γ' ← ∅
foreach G = (V, E, C, L) ∈ Γ do
    C' ← C
    foreach c_u ∈ C do
        veracity ← DetectVeracity(c_u)
        if veracity = Fake then
            V ← V \ {u}
            E ← E \ {(u, v)|(u, v) ∈ E}
            C' ← C' \ {c_u}
            L ← L \ {l_u}
    G' = (V, E, C', L)
    Γ' = Γ' ∪ {G}
return Γ'
```

1) the content similarity between the nodes given the membership level m of the content of a node c_u to a topic t_k, and
2) the proximity of nodes to a core point p that defines a geographic area A.

To compute the content similarity, we first use a topic modeling algorithm to extract k topics T within our graphs. For extracting topics, we can employ any topic modeling algorithm, e.g., LDA, NMF, OLDA, etc. In our implementation, we choose OLDA. Then we determine the membership of a node's content c_u to belong to a topic $t_j \in T$ ($j = \overline{1,k}$) using the similarity $sim(c_u, t_j)$ based on the topic probability distribution given by the employed topic model. Given the similarity $sim(c_u, t_j)$ between a node's content and a topic, if the similarity is over a given threshold ε_c ($sim(c_u, t_j) \leq \varepsilon_c$) then c_u belongs to topic t_j. Based on the threshold, the same node's content can belong to multiple topics. Using the topics T and the graph Γ', we construct the topic graphs Θ.

Algorithm 3 presents the topic graphs extraction. The algorithm receives as input the undirected clean graphs Γ', the number of topics k, and the similarity threshold ε_c. The output is the topic graphs Θ. Line 1 initializes Θ. Lines 2 to 5 concatenates the content of all the graphs G and extract k topics stored in T. The $Topics(D, k)$ (Line 5) function can be implemented using any topic modeling algorithm. In our implementation we choose OLDA, but any other topic modeling algorithms can be used [45], [46], i.e., NMF, LSI, etc. We iterate through each graph and each topic to determine the membership of the content of a node to a topic using the similarity and create a topic graph G' which is added to the topic graphs Θ (Lines 6 to 19). At the end of the iteration, the topic graphs Θ are returned (Line 20).

The proximity of a node u to a core point p is computed using its geolocation coordinates l_u. The intuition behind this assumption is based on the fact that nodes that are near each other will fall within the same geographic area A. Each geographic area A represents a cluster. The area A is given by a core point p and a radius ε_l that specifies the cluster's maximum extent. Thus, the proximity of a node u is computed as distance $\delta(l_u, p)$ between

Algorithm 3: $TopicGraphs$ - Topic Graphs Extraction.

```
   Input  : the undirected clean graphs Γ'
            the number of topics k
            the similarity threshold ε_c
   Output: the topic graphs Θ
1  Θ ← ∅
2  D ← ∅
3  foreach G = (V, E, C, L) ∈ Γ' do
4      D ← D ∪ C
5  T ← Topics(D, k)
6  foreach t ∈ T do
7      V' ← ∅
8      E' ← ∅
9      C' ← ∅
10     L' ← ∅
11     foreach G = (V, E, C, L) ∈ Γ' do
12         foreach c_u ∈ C do
13             if sim(c_u, t) ≥ ε_c then
14                 V' ← V' ∪ {u}
15                 E' ← E' ∪ {(u, v)|(u, v) ∈ E}
16                 C' ← C' ∪ {c_u}
17                 L' ← L' ∪ {l_u}
18     G' = (V', E', C', L')
19     Θ ← Θ ∪ {G'}
20 return Θ
```

its geolocation l_u and the core point p. If $\delta(l_u, p) \leq \varepsilon_l$ then $u \in A$. The area A and the core point p are determined using a data clustering algorithm, e.g., DBSCAN [47]. Using the content similarity and the proximity of nodes, we create geolocation-content based communities $\Sigma = (V_\Sigma, E_\Sigma, C_\Sigma, L_\Sigma)$. Within the same area, there can be multiple communities.

Algorithm 4 presents the construction of the geolocation-content based communities. The algorithm receives as input the topic graph $\theta \in \Theta$ and the proximity threshold ε_l and outputs the communities Σ. Line 1 initializes Σ. Lines 2 to 5 extract all the geolocations from θ and determine the areas A for which the radius is equal to ε_l. We iterate through each graph and each area to construct the communities Σ using the condition $\delta(l_u, p) \leq \varepsilon_l$ (Lines 6 to 19). At the end of the iteration, the geolocation-content based communities Σ are returned (Line 20).

Algorithm 5 presents ContCommRTD, the solution to our problem. The algorithm receives as input the keywords and hashtags list H, the number of topics k, the similarity threshold ε_c, and the proximity threshold ε_l. Firstly, the social graph Γ is build using H and Algorithm 1 (Line 2). Secondly, the graph Γ' is constructed by removing from Γ all the nodes that contain misinformation (Line 3). Thirdly, the topic graphs are extracted (Line 4). Lastly, the geolocation-content based communities Σ are determined for each topic graph and returned (Lines 5 to 7).

IV. System Description

The architecture of the proposed system *ContCommRTD*, which implements Algorithm 5, is presented in Fig. 1.

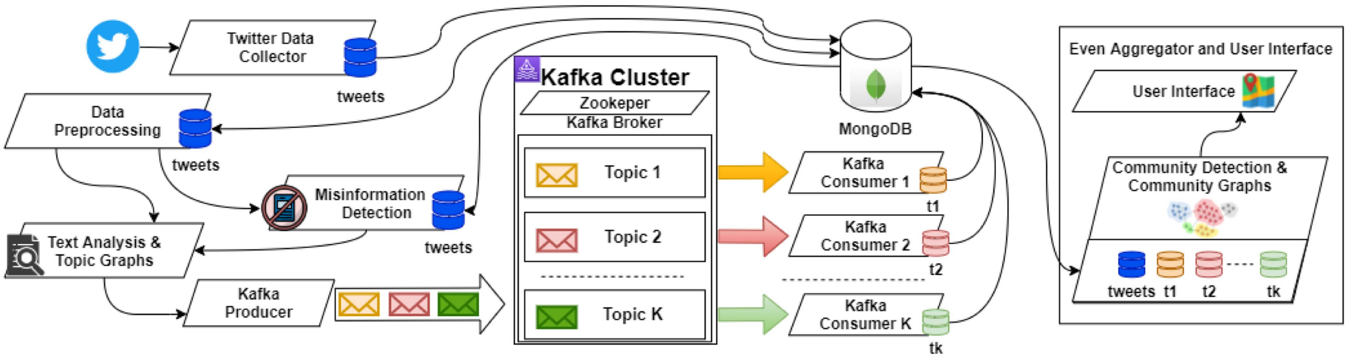


Fig. 1. Architecture of *ContCommRTD* (database colors denote the collections used by each module).

Algorithm 4: $CommunityGraphs$ - Geolocation-content Based Communities Extraction.

Input : the undirected topic graph θ
the proximity threshold ε_l
Output: the communities $\Sigma' = (V_\Sigma, E_\Sigma, C_\Sigma, L_\Sigma)$

$\Sigma' \leftarrow \emptyset$
$P \leftarrow \emptyset$
foreach $(V, E, C, L) \in \theta$ **do**
  $P \leftarrow P \cup L$
$A \leftarrow Areas(P, \varepsilon_l)$
foreach $p \in A$ **do**
  $V' \leftarrow \emptyset$
  $E' \leftarrow \emptyset$
  $C' \leftarrow \emptyset$
  $L' \leftarrow \emptyset$
  foreach $(V, E, C, L) \in \theta$ **do**
    foreach $l_u \in L$ **do**
      if $\delta(l_u, p) \leq \varepsilon_l$ **then**
        $V' \leftarrow V' \cup \{u\}$
        $E' \leftarrow E' \cup \{(u, v) | (u, v) \in E\}$
        $C' \leftarrow C' \cup \{c_u\}$
        $L' \leftarrow L' \cup \{l_u\}$
  $G' = (V', E', C', L')$
  $\Sigma' \leftarrow \Sigma' \cup \{G'\}$
return Σ'

Algorithm 5: *ContCommRTD* - Geolocation Community Raphs Extraction.

Input : the keywords and hashtags list H
the number of topics k
the similarity threshold ε_c
the proximity threshold ε_l
Output: the communities $\Sigma = (V_\Sigma, E_\Sigma, C_\Sigma, L_\Sigma)$

$\Sigma \leftarrow \emptyset$
$\Gamma \leftarrow SocialGraph(H)$
$\Gamma' \leftarrow MisinformationDetection(\Gamma)$
$\Theta \leftarrow TopicGraphes(\Gamma', k, \varepsilon_c)$
foreach $G = (V, E, C, L) \in \Theta$ **do**
  $\Sigma \leftarrow \Sigma \cup CommunityGraphs(G, \varepsilon_l)$
return Σ

A. Twitter Data Collector

This module is used to fetch Twitter data in real-time, using the Twitter Developer API and it implements Line 7 from Algorithm 1. To increase the Twitter read-limit rate cap, we use a Premium API subscription. This module collects tweets based on the chosen type of hazard, e.g., extreme hydrological hazards associated with water-related events. The initial selection is done by searching tweets that contain specific words. For this purpose, we created a meta-knowledge dictionary as follows. For each new type of hazard, we add a new entry containing a list of top-k keywords and hashtags for the chosen hazardous event. To extract the keywords, we employ TLATR [48] using the following pipeline:

1) extract the topics and keywords including hashtags
2) label the topics.

When users search for new events, they will select from a list of topics' labels and the system will automatically load all the topic's keywords and hashtags, which in turn are used to filter the stream of tweets. The collected tweets are stored in *tweets* collection in a NoSQL Document-Oriented Distributed Database Management system, i.e., MongoDB. We choose MongoDB because benchmarks show that this system is fast, reliable, and offers good performances when dealing with textual data [49], [50].

Using these filters, we collect the tweet and its retweets and replies. For a tweet, we store the following information: id, language, retweet flag, creation date, user information, geolocation information, and text. For a retweet, we also keep the retweeter's coordinates and location. Each retweet information is also updated in real-time with the original tweet's data. The same information is also kept for replies. We use geolocation as a sharding key to distribute the data between different sites and improve querying.

B. Data Preprocessing and Geolocation Enhancement

This module implements constructs the social graph Γ implemented by Algorithm 1.

1) Data Preprocessing.: We extract the textual content of all the Tweets from the database and apply the following preprocessing steps to extract a *clean text:*

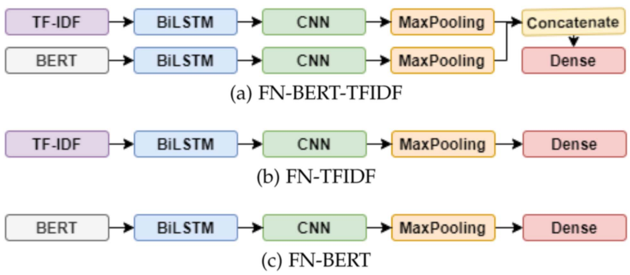


Fig. 2. Deep learning architectures.

1) remove user mentions (terms that start with @), http links, punctuation, double spacing, and numbers from the text;
2) Transform the text to lowercase;
3) Extract terms through tokenization;
4) Remove stop words and tokens with a length smaller than 3 characters;
5) Extract the stem of the remaining terms using the Porter Stemmer [51] to reduce the number of terms by removing inflected and derived words, thus minimizing the vocabulary.

Using the clean text and the normalized Term Frequency-Inverse Document Frequency $TFIDF_n$, we built the document-term matrix $W = [w_{ij}]$ needed by the analysis algorithm, where $w_{ij} = TFIDF_n(t_{ij}, d_i, D), t_{ij}$ is a term appearing in document d_i, and D is the entire collection of tweets. We use the *scikit-learn* [52] $TFIDF$ implementation.

2) Analyzing the Geolocation of Tweets: We utilize several techniques to obtain the geolocation information of a tweet, retweet, or reply. The most basic approach was to get the coordinates of a tweet. For this to work, the user must have the location turned on. There are various studies (e.g., [53]) that show that only a relatively small percentage of users use this feature. However, we observe that a large majority of the collected hazard-related tweets are either geotagged or contain location information in the text, e.g., "Heavy rain in Masjid Al Haram". A tweeter that contains a Place has 4 pairs of coordinates that define the area for that Place. If the tweet doesn't have a geolocation tag either, we apply Named Entity Recognition to identify possible locations from the text. Once identified, we use GeoNames [54] to get the coordinates. This is an improvement over many existing real-time systems that solely rely on geolocation information embedded within the tweet to determine events [22], [23].

C. Misinformation Detection

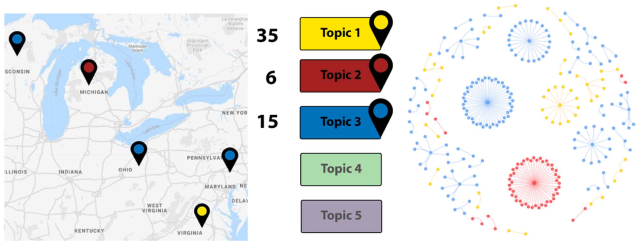
This module is used to filter and mark any tweets that spread misinformation, i.e., Fake News (FN). This module implements Algorithm 2. As the Twitter datasets that contain disaster-related events have usually a limited size, we use Transfer Learning techniques to transfer knowledge gained from these larger but more generic Twitter datasets to misinformation detection on disaster-related datasets. For the task of misinformation detection, we propose a new Deep Learning architecture, FN-BERT-TFIDF (Fig. 2(a)), that receives as input the TF-IDF vector for a tweet as well as tweet embedding extracted with BERT. For each input, the model contains one BiLSTM layer, a CNN layer, and a MaxPooling layer. We choose a BiLSTM layer because it enables the network to use both previous and future elements by looking forward and backward in the words' sequence. We use a CNN layer to create new features using the convolution operation between the text window and every distinct filter. We use a MaxPooling layer to decrease the size of the feature channels by grouping elements into fixed-length sequences and choosing only the feature with the maximum value. The output of the MaxPooling layers is concatenated and sent as input to the classification layer. To test the efficiency of our model, we perform ablation testing and compare the results of the FN-TFIDF-BERT model with the results obtained when only TF-IDF, i.e., FN-TFIDF (Fig. 2(b)), or BERT, i.e., FN-BERT (Fig. 2(c)), is used as input. After a tweet passes through the detection model, its record in MongoDB is updated with a new field that encodes its veracity. If a tweet is detected as fake, then it will not be used in the next modules. For implementing the Deep Learning models, we use the *Keras* interface of the *TensorFlow* [55]. For building the BERT embeddings, we use *simpletransformers* with the *HuggingFace* [56] BERT model.

D. Text Analysis and Topic Graphs

This module implements Algorithm 3 to extract the topic graphs. To Analyze the tweets and determine the context regarding the hazard discussed in the text, we use Online Latent Dirichlet Allocation (OLDA) [57]. OLDA is a generative statistical model used for topic modeling that groups together terms that are syntactically different but have a similar meaning and represent the same concepts. The algorithm determines for each tweet a specific topic by calculating the similarity between the tweet and all of the topics. Thus, OLDA assigns a tweet to a mixture of topics, i.e., each tweet is a combination of one or more topics. The model is built in real-time, as more tweets are added to the database, the model is retrained with the new information. We use a 50% threshold (i.e., ε_c) to determine if a tweet is relevant to a topic as a tweet can belong to multiple topics. We use the *gensim* [58] OLDA implementation.

E. Kafka Modules

The *Kafka Producer* module takes $<$ tweet, topic $>$ pairs and sends them to the *Kafka Consumers*. *Kafka Cluster* is used to create an environment that assures the distributed exchange of messages between the *Kafka Producer* and the *Kafka Consumers*. It contains the *Kafka Broker* and the *Zookeeper* service. The job orchestration and cluster topology are done by the *Zookeeper* service, a distributed coordination service for distributed applications [59]. The *Kafka Broker*'s main role is to act as a transition channel. It takes messages in the form of $<$ tweet, topic $>$ pairs and delivers them to the right *Kafka Consumer*. Thus, the receiving of data by the *Kafka Consumers* is done in parallel in real-time in a non-blocking way. Each *Kafka Consumer* stores messages in their own topic-dependent collections within MongoDB and stores $<$ topic, [(tweet_ids, geolocation)]$>$ pairs, i.e., a topic and a list of all the related



(a) Google map view (b) Subscriptions (c) Communities per topic

Fig. 3. User interface.

tweets ids with their geolocation. The geolocation is used for sharding.

F. Community Graphs and User Interface

This module is used to present in real-time to users one or more topics of interest from one or more hazard related fields on Google Maps (Fig. 3(a)). Users can choose and subscribe to desired topics (Fig. 3(b)) and can monitor a specific area on the map just by searching for it. For each subscribed topic, the user sees on the map colored pins updated automatically in almost real-time. A user can also view the formed communities based on the chosen topics (Fig. 3(c)).

Each pin represents a community for a specific topic within the social graph. To form these communities, we use Algorithm 4 with DBSCAN (Density Based Spatial Clustering of Applications with Noise) [47], a density-based clustering algorithm. DBSCAN computes the density of points (core points) p around which we form the area A. Thus, the clustering algorithm measures density as the number of points within the radius ε_l of a point from the analyzed dataset. It can estimate the connected components of the λ-density level set $x : f(x) \geq \lambda$ given n samples from an unknown density f [60].

The user has access to the <topic, tweets> pairs. The topic is extracted from the MongoDB based on the user's subscribed topics. By matching the tweet_ids presented in the *tweets* and *topic* collections, we extract all the geographical coordinates and group them together using DBSCAN. Thus, the same topic can appear multiple times on the map in different geographic locations. For these geographic locations, we present to the user the latest information as the tweets are updated in real-time by the *Twitter Data Collector* module. This component employs strong synchronization, in order to retrieve information from the database in near real-time.

If a user unsubscribes from a topic, then the corresponding pins are removed from the map. We use timestamped matching queries to retrieve the relevant information. The pins for the topics that have not been updated in more than 24 h are removed automatically from the map. Their topics are kept in *tweets* collection for retraining and improving the performance of the OLDA model.

G. Design Challenges

With real-time data stream collection systems, it is important to take into account that the system may be disconnected from the service. To mitigate this challenge, we implemented a background process checker that verifies every 30 seconds that the *Twitter Data Collection* is working. Also, Twitter is limiting the number of tweets that can be collected in an hour. We address this challenge by using filters to select only relevant tweets.

To build the topic graph, we use OLDA. The reason behind choosing OLDA, instead of LSI or NMF, is that OLDA is a probabilistic model while NMF and LSI are matrix factorization and multivariate analysis techniques. With the increase in the volume of collected data, the runtime performance of OLDA can decrease. Thus, we systematically start in the background a new OLDA training process each hour. When a training process is finished, the new model is uploaded to the application, and the old one is removed from production and archived. Furthermore, we also use the same approach to fine-tune the misinformation model to improve performance. We use the same strategy for building the community graph with DBSCAN as we do for constructing the topic graph with OLDA. We use DBSCAN as we do not need to know apriori the number of clusters. We use sharding for our database to increase localization, load balancing, and querying through geodistribution. For all the collections, we use the geolocation coordinates as the shard key. We also use replication to create three member replica sets for each shard. Through this mechanism, we increase data availability, add fault and partition tolerance, and eliminate the single point of failure. These design choices are also used to seamlessly scale the datasets horizontally.

The processes that create backups for our models (i.e., the topic models using OLDA, the FN-BERT-TFID fake news detection models, and the community graphs models using DBSSCAN) create a sequence of snapshots that can be analyzed either individually or in bulks using time frames, improving our understanding of the evolution of hazards for different periods. Analyzing multiple models using a sequence of snapshots improves the detection of changing communities over time. Thus, the proposed system is always aware of the changes that can dynamically appear in our communities, managing to adapt in real-time to the new data that it receives. Also, these snapshots provide a recovery mechanism for the models in case of unexpected failures.

V. Evaluation

In this section, we present the experimental results of our proposed system, ContCommRTD. We first focus on the ablation testing for the misinformation detection models. Secondly, we present two use cases of ContCommRTD for disaster reporting:

1) Hydrological Hazards and
2) COVID-19 Infection Hazards.

We conclude this section with ContCommRTD scalability tests.

TABLE I
EXAMPLE OF FILTER KEYWORDS (SUBSET)

Keywords flood	flowage, rain, precipitation, floodplain, groundwater, overflow, deluge water level, water flow, rainfall, inundation, torrent, groundwater flood tsunami, torential, costal flooding, costal storm, river flooding, hurricane, ...
Hashtags flood	#flood, #precipitation, #rainfall, #deluge, #torrent, #inundation, #rain #floods, #waterlevel, #hurricane, #tornado, #torential, #storm, #flowage, ...
Keywords Covid-19	coronavirus, corona, COVID, COVID-19, pandemic, quarantine, lockdown, corona virus, hand sanitizer, infection, wash your hands, mask, personal protective equipment, covid quarentined, intensive care ...
Hashtags Covid-19	#covid, #sarscov2, #corona, #vaccine, #stayhealthy, #stayathome, #SARSCoV2 #viruscorona, #quarentined, #quarentinelife, ...

TABLE II
MISINFORMATION ABLATION RESULTS

Dataset	Model	Accuracy	Precision	Recall	F1 Score
LIAR	FN-TFIDF	56.01 ± 0.33	55.81 ± 0.52	55.85 ± 0.54	55.83 ± 0.53
	FN-BERT	58.75 ± 0.44	57.97 ± 0.50	57.09 ± 0.46	57.53 ± 0.48
	FN-BERT-TFIDF	$\mathbf{60.92 \pm 0.59}$	60.65 ± 0.90	60.66 ± 0.97	60.65 ± 0.93
	LSTM [70]	$\sim$58	N/A	N/A	N/A
Covid-19	FN-TFIDF	81.93 ± 0.37	78.48 ± 0.64	76.90 ± 0.11	77.68 ± 0.19
	FN-BERT	86.86 ± 0.07	86.86 ± 0.07	86.86 ± 0.07	86.86 ± 0.07
	FN-BERT-TFIDF	$\mathbf{87.92 \pm 0.04}$	87.91 ± 0.02	87.92 ± 0.03	87.82 ± 0.02
	VAE + MLP [68]	N/A	N/A	N/A	85.98 ± 0.10

The bold marks the best performing model for a given dataset.

A. Data Collection

We collect two datasets for the two use cases. First, for the Hydrological Hazards use case we collect 356 483 Tweets using the hydrological data dictionary and store them in the distributed MongoDB database. This dataset is also used to train the first OLDA model and to evaluate the algorithm's runtime performance and topic quality for the first use case. Second, for the COVID-19 Infection Hazards we collect 50 230 and present the community detection results. Table I presents a subset of the used keywords and hashtags for both hydrological and COVID-19 use cases. The meta-dictionary is created using the website https://best-hashtags.com/ (Accessed on the 21st of April 2023) together with lists provided by experts from the filed of hydrology (for the hydrological hazards use case) and medicine (for the COVID-19 use case).

B. Evaluation Metrics

1) Misinformation Detection: To evaluate the quality of the Deep Learning architectures employed for Misinformation Detection, we use Accuracy, Precision, and Recall.

2) Topic Modeling: To evaluate the quality of the topic model, we employ perplexity and topic coherence [61]. Perplexity is a measure that determines how well a probability model predicts a sample [62]. A low perplexity denotes that the distribution predicts correctly the sample. For topic modeling, an algorithm that achieves a low perplexity indicates that it fit the data better. Topic Coherence measures the human-interpretability of a topic [63]. We use the C_V [64] to measure the coherence of our topics. For perplexity, we use *gensim*'s *log_perplexity()* implementation, while for C_V we use *palmetto* [64].

3) Community Detection: To evaluate the quality of the communities, we employ Davies-Bouldin [65], Calinski-Harabasz [66], and Silhouette [67]. The Davies-Bouldin score measures the separation between clusters by computing the ratio between within-cluster distances and between-cluster distances which determines the average similarity of each cluster with its most similar cluster. The Davies-Bouldin score bounded in the range $[0, 1]$, whit a score closer to 0 showing better separation. The Calinski-Harabasz score measures how well are the clusters defined by computing the ratio between the within-cluster dispersion and the between-cluster dispersion. A higher Calinski-Harabasz shows core shows clusters that are dense and well separated. Unfortunately, the Calinski-Harabasz score is not bounded. The Silhouette score is another measurement for determining how well are the clusters defined by computing the mean intra-cluster distance and the mean nearest-cluster distance. The Silhouette is defined in $[-1, 1]$ range, with scores closer -1 for incorrect clusters, scores closer to $+1$ for highly dense clusters, and scores around 0 for overlapping clusters.

C. Misinformation Detection

To test the proposed models, we use two publicly available dataset: COVID-19 dataset [68] and LIAR [69] using 2 labels, i.e., fake or real. The LIAR dataset contains approximately 12.8 K human annotated short statements collected over the time span of a decade (primarily from 2007–2016) using POLITIFACT.COM's API. The COVID-19 rumor datasets contain $6,834$ manually annotated rumors from news and tweets primarily collected from January to April 2020. Following the data cleaning and preprocessing step, we have the following class distribution: 1) LIAR dataset: 2 046 True and 10 709 not True (i.e., 2 501 false, 2 622 half-true, 1043 pants-fire, 2 446 mostly-true, and 2 097 barely-true), 2) COVID-19 datasets: 1 434 True, 3 461 False, and 1 254 Unverified.

For the experiments, we use a 70%–30% train-test split with random shuffle, while maintaining the class ratio for the two sets. We apply k-folds cross-validation with $k = 10$. The BiLSTM network has 256 units, each unit with a dropout of 0.2. For the CNN layer, we use 64 filters and a kernel size of 128. We use a 5 000-dimensions TF-IDF vector and the pretrained 1 024-dimension BERT transformer from *HuggingFace* [56], i.e., *bert-large-uncased*. We observed that FN-BERT-TFIDF outperforms the other two models on both datasets (Table II).

On both datasets, we observe that FN-TFIDF and FN-BERT obtain worse results individually than when put together to form the FN-BERT-TFIDF model. Thus, FN-BERT-TFIDF model obtains an accuracy of 60.92% on LIAR and 87.92% on Covid-19 datasets because it leverages the local and global statistic information given by the TFIDF embedding and context information given by the BERT word embeddings. We also observe that the proposed model outperforms existing models on these datasets.

D. Topic and Communities Detection

When training OLDA, we need to initialize two parameters:

1) *Alpha* the document-topic density (a larger value means that a document contains a larger number of topics), and
2) *Beta* the topic-word density (a larger value means that more words are considered to belong to the same topic).

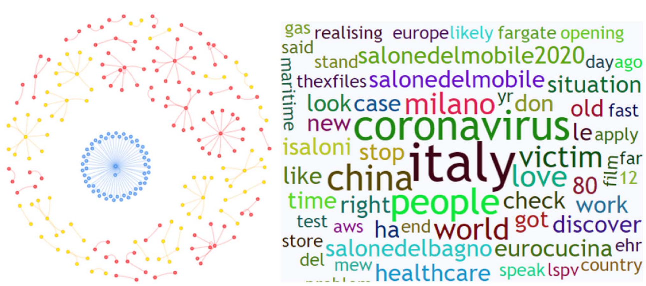
We initialize these values to "auto" in order to learn them automatically from the corpus. After training the OLDA model 10 times on the initial corpus, we obtained an average runtime of 17 min.

TABLE III
EXAMPLE OF OLDA TOPICS FOR HYDROLOGICAL HAZARDS

Topic	Keywords and importance					
1	rain 0.133	wind 0.039	temp 0.025	disaster 0.022	humidity 0.015	weather 0.011
2	storm 0.088	june 0.013	power 0.008	groundwater 0.008	look 0.007	together 0.007
3	rain 0.026	disaster 0.016	time 0.015	people 0.012	flood 0.011	need 0.010
4	disaster 0.057	local 0.033	authority 0.026	must 0.021	meeting 0.018	region 0.012
5	hurricane 0.044	water 0.034	overflow 0.021	help 0.019	time 0.018	authority 0.018
6	thunderstorm 0.053	severe 0.046	warning 0.044	storm 0.032	tornado 0.019	county 0.018

TABLE IV
EXAMPLE OF MATCHING TWEETS TO TOPICS

Tweet	Topic 1	2	3	4	5	6
After a hurricane, a guy found this pittie on the roof of a submerged car waiting for help	0.02	0.02	0.02	0.02	0.90	0.021
Happening now: A 33,000 litres of truck laden with diesel was prevented from causing another national disaster by Officers and men of Federal Fire Service at Ojuelegba bridge. Situation under control and normalcy restored.	0.0	0.0	0.0	0.96	0.0	0.0



(a) Communities (b) Word cloud for "blue" community

Fig. 4. COVID-19 infection hazards.

1) Use Case 1: Hydrological Hazards: We extract 6 topics (Table III) and evaluate the quality of the model using perplexity, obtaining a score of 7.83 for the entire dataset. This score shows that OLDA manages to predict well the data sample from our dataset. Finally, we compute the C_V score to determine topic coherence and human-interpretability. For the OLDA model train on the initial corpus, we obtain a $C_V = 0.48$, meaning that the topics are readable by users. The evaluation of the community graphs shows a good separation and well-bounded clusters. The scores we obtain are as follows: 0.0286 Davies-Bouldin, $\sim 3.60e9$ Calinski-Harabasz, and 0.9758 Silhouette.

Table IV presents some examples of tweet-topic distribution. In our examples, we observe that the first tweet is labeled with topic 5 where the main keyword is "hurricane", while the second tweet is correlated with topic 4 where the main keyword is "disaster". These two examples are highly correlated with the extracted topics from Table III.

2) Use Case 2: COVID-19 Infection Hazards: As in the first use case, we train an OLDA model to create geolocation-content based communities on COVID-19 related topics. Fig. 4(a) depicts the communities for three major Coronavirus related topics, while Fig. 4(b) shows the word cloud for the "blue" topic from Fig. 4(a) (#1 in Table V). We extract 3 topics (Table V) for which we obtain a perplexity score of 9.01 and a C_V score

TABLE V
EXAMPLE OF OLDA TOPICS FOR COVID-19 HAZARDS

Topic	Keywords and importance					
1	cases 0.090	home 0.080	intensive 0.080	care 0.079	#coronavirus 0.072	quarantine 0.070
2	coronavirus 0.116	covid-19 0.078	hospital 0.035	virus 0.035	wuhan 0.035	chinese 0.034
3	#coronavirus 0.198	outbreak 0.119	health 0.090	vaccine 0.080	spread 0.072	died 0.069

TABLE VI
RUNTIME IN SECONDS: WRITE VS. READ OPERATIONS

		No. Clients			
		100	1 000	10 000	100 000
No. Tweets	100	0.13 ± 0.01	0.54 ± 0.01	4.49 ± 0.16	44.10 ± 0.30
	1 000	0.36 ± 0.01	0.55 ± 0.01	4.43 ± 0.05	43.67 ± 1.21
	10 000	3.45 ± 0.11	3.59 ± 0.03	4.80 ± 0.02	48.05 ± 0.05
	100 000	33.37 ± 0.48	34.60 ± 0.07	34.83 ± 0.77	87.67 ± 1.04

of 4.8 for the entire dataset. We obtain the following score for the community graphs: 0.0197 Davies-Bouldin, $\sim 2.62e9$ Calinski-Harabasz, and 0.9853 Silhouette. By evaluating these scores, we observe that we obtain well-separated and bounded clusters. By changing the number of topics to be detected, we can create an improved view regarding subtopics, thus bringing another layer of granularity to our analysis and the end users.

E. Scalability

To test the scalability of the proposed system, we simulate multiple scenarios that simultaneously use two types of processes in a pseudo-distributed environment. The first type of process connects to the MongoDB database, inserts a new tweet in the *tweets* collections, and updates its topic-dependent collection. The second type of process simulates multiple clients that count all the tweets present at a given moment for a topic of interest. This set of experiments are run on an IBM System x3550 M4 with 64 GB of RAM and an Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50 GHz with 40 cores.

Table VI presents the results for 10 executions. For a tiny number of write and read operations, i.e., the number of tweets writes $\leq$1 000 and the number of clients reads $\leq$1 000, the system is stable and handles the burst of requests in under a second. In this case, the maximum response time of 0.55 seconds is registered for 1 000 write and 1 000 read operations. As the number of operations increases, the system handles the request in under 1.5 minutes. We note that the probability of receiving a high number of tweets and having a large number of client requests, even during a disaster, is very low.

VI. CONCLUSION

In this paper, we present ContCommRTD, a new distributed system that determines geolocation-content based communities depending on the topics of interest and user geolocation, and takes into account misinformation on social networks. Moreover, we have also shown how our system can be applied to track the evolution of different hazards, provided that there are citizens who post this information on social media. This is of significant importance, especially for government organizations that can subscribe to topics of interest in a specific geographical area or can subscribe to receive information on a topic regardless of the location. Our system also provides an interactive graphical interface where a user can select what are the topics of interest. On the user map, the notifications within the relevant topics will appear colored. All the notifications that came from the detection of other information apart from the chosen topics are displayed in gray color. Furthermore, our approach can be generalized for other types of hazards or social events if new terms are added to the dictionary used to collect specific tweets. As tweets collected from unreliable or unchecked sources may spread misinformation, we consider detecting these tweets and removing them from our active communities. For this task, we propose FN-BERT-TFIDF, a new Deep Learning BERT-based model. Moreover, the proposed system scales well as the number of read and write operations increases.

In future work, we plan to add more information in order to analyze human behaviors during disasters, as follows. Between the filtering and preprocessing process, the tweet could undergo a more detailed analysis, such as domain-specific rule-based filtering and disambiguation [16], [71], and text-based sentiment analysis [72], [73], [74]. For clustering and extracting the communities, we plan to employ novel density-based algorithms such as HDBSCAN [75] or DenLAC [76]. Furthermore, we aim to integrate network immunization strategies to improve the misinformation detection module with mitigation methods [77], [78].

References

[1] M. Walther and M. Kaisser, "Geo-spatial event detection in the twitter stream," in *Proc. Eur. Conf. Inf. Retrieval*, 2013, pp. 356–367.

[2] E. Starkey, G. Parkin, S. Birkinshaw, A. Large, P. Quinn, and C. Gibson, "Demonstrating the value of community-based ('citizen science') observations for catchment modelling and characterisation," *J. Hydrol.*, vol. 548, pp. 801–817, 2017.

[3] G. Cervone, E. Sava, Q. Huang, E. Schnebele, J. Harrison, and N. Waters, "Using twitter for tasking remote-sensing data collection and damage assessment: 2013 boulder flood case study," *Int. J. Remote Sens.*, vol. 37, no. 1, pp. 100–124, 2015.

[4] Z. Li, C. Wang, C. T. Emrich, and D. Guo, "A novel approach to leveraging social media for rapid flood mapping: A case study of the 2015 south carolina floods," *Cartogr. Geographic Inf. Sci.*, vol. 45, no. 2, pp. 97–110, 2017.

[5] L. Smith, Q. Liang, P. James, and W. Lin, "Assessing the utility of social media as a data source for flood risk management using a real-time modelling framework," *J. Flood Risk Manage.*, vol. 10, no. 3, pp. 370–380, 2015.

[6] J. F. Rosser, D. G. Leibovici, and M. J. Jackson, "Rapid flood inundation mapping using social media, remote sensing and topographic data," *Natural Hazards*, vol. 87, no. 1, pp. 103–120, 2017.

[7] A. Guille and C. Favre, "Event detection, tracking, and visualization in twitter: A mention-anomaly-based approach," *Social Netw. Anal. Mining*, vol. 5, no. 1, pp. 1–18, 2015.

[8] J. Fohringer, D. Dransch, H. Kreibich, and K. Schröter, "Social media as an information source for rapid flood inundation mapping," *Natural Hazards Earth Syst. Sci.*, vol. 15, no. 12, pp. 2725–2738, 2015.

[9] B. Wang and J. Zhuang, "Rumor response, debunking response, and decision makings of misinformed twitter users during disasters," *Natural Hazards*, vol. 93, no. 3, pp. 1145–1162, 2018.

[10] C.-O. Truică and E.-S. Apostol, "MisRoBÆRTa: Transformers Versus Misinformation," *Mathematics*, vol. 10, no. 4, pp. 1–25, 2022.

[11] N. Garg, *Apache Kafka*. Birmingham, U.K.: Packt Publishing, 2013.

[12] G. Hesse, C. Matthies, and M. Uflacker, "How fast can we insert? An empirical performance evaluation of apache kafka," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst.*, 2020, pp. 641–648.

[13] K. Chodorow, *MongoDB: The Definitive Guide*. Sebastopol, CA, USA: O'Reilly Media, 2010.

[14] B. Hamoui, M. Mars, and K. Almotairi, "FloDusTA: Saudi tweets dataset for flood, dust storm, and traffic accident events," in *Proc. 12th Lang. Resour. Eval. Conf.*, 2020, pp. 1391–1396.

[15] A. Hernandez-Suarez et al., "Using twitter data to monitor natural disaster social dynamics: A recurrent neural network approach with word embeddings and kernel density estimation," *Sensors*, vol. 19, no. 7, 2019, Art. no. 1746.

[16] W. Lukasiewicz, K. Teymourian, and A. Paschke, "A rule-based system for monitoring of microblogging disease reports," in *The Semantic Web: ESWC 2014 Satellite Events*. Berlin, Germany: Springer, 2014, pp. 401–406.

[17] C. Ferner, C. Havas, E. Birnbacher, S. Wegenkittl, and B. Resch, "Automated seeded latent dirichlet allocation for social media based event detection and mapping," *Information*, vol. 11, no. 8, 2020, Art. no. 376.

[18] M. Riegler, K. Pogorelov, L. Hassan, N. Ahmad, and N. Conci, "Natural disasters detection in social media and satellite imagery: A survey," *Multimedia Tools Appl.*, vol. 78, no. 22, pp. 31267–31302, 2019.

[19] X. Huang, Z. Li, C. Wang, and H. Ning, "Identifying disaster related social media for rapid response: A visual-textual fused CNN architecture," *Int. J. Digit. Earth*, vol. 13, no. 9, pp. 1017–1039, 2019.

[20] R. I. Jony, A. Woodley, and D. Perrin, "Flood detection in social media images using visual features and metadata," in *Proc. IEEE Conf. Digit. Image Comput. Techn. Appl.*, 2019, pp. 1–8.

[21] C. Arachie, M. Gaur, S. Anzaroot, W. Groves, K. Zhang, and A. Jaimes, "Unsupervised detection of sub-events in large scale disasters," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 354–361.

[22] C. Zhang et al., "TrioVecEvent: Embedding-based online local event detection in geo-tagged tweet streams," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 595–604.

[23] C. Zhang et al., "GeoBurst : Effective and real-time local event detection in geo-tagged tweet streams," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 3, pp. 1–24, 2018.

[24] C. Loynes, J. Ouenniche, and J. D. Smedt, "The detection and location estimation of disasters using twitter and the identification of non-governmental organisations using crowdsourcing," *Ann. Operations Res.*, vol. 308, pp. 339–371, 2020.

[25] R. Suwaileh, M. Imran, T. Elsayed, and H. Sajjad, "Are we ready for this disaster? Towards location mention recognition from crisis tweets," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 6252–6263.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 4171–4186.

[27] M. Dou, Y. Wang, Y. Gu, S. Dong, and Y. D. M. Qiao, "Disaster damage assessment based on fine-grained topics in social media," *Comput. Geosciences*, vol. 156, 2021, Art. no. 104893.

[28] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, no. 6, 2004, Art. no. 066111.

[29] T. Ma, Y. Zhao, H. Zhou, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, "Natural disaster topic extraction in sina microblogging based on graph analysis," *Expert Syst. Appl.*, vol. 115, pp. 346–355, 2019.

[30] M.-T. Nguyen, T.-T. Nguyen, A. Kitamoto, and V.-H. Nguyen, "Exploiting social networks as a live mass media channel during disasters for reactions," *Int. J. Artif. Intell. Tools*, vol. 30, no. 05, 2021, Art. no. 2150024.

[31] V.-I. Ilie, C.-O. Truică, E.-S. Apostol, and A. Paschke, "Context-aware misinformation detection: A benchmark of deep learning architectures using word embeddings," *IEEE Access*, vol. 9, pp. 162122–162146, 2021.

[32] B. Palani, S. Elango, and V. V. Ku, "CB-Fake: A multimodal deep learning framework for automatic fake news detection using capsule neural network and BERT," *Multimedia Tools Appl.*, vol. 81, no. 4, pp. 5587–5620, 2021.

[33] S. Sharma, M. Saraswat, and A. K. Dubey, "Fake news detection on twitter," *Int. J. Web Inf. Syst.*, vol. 18, no. 5/6, pp. 388–412, 2022.

[34] C.-O. Truică, E.-S. Apostol, and A. Paschke, "Awakened at CheckThat! 2022: Fake news detection using BiLSTM and sentence transformer," in *Proc. Work. Notes Conf. Labs Eval. Forum*, 2022, pp. 749–757.

[35] C.-O. Truică and E.-S. Apostol, "It's all in the embedding! fake news detection using document embeddings," *Mathematics*, vol. 11, no. 3, pp. 1–29, 2023.

[36] C.-O. Truică, E.-S. Apostol, and P. Karras, "DANES: Deep neural network ensemble architecture for social and textual context-aware fake news detection," *Knowl.-Based Syst.*, vol. 294, pp. 1–13, 2024.

[37] D. Singh, S. Shams, J. Kim, S.-J. Park, and S. Yang, "Fighting for information credibility: An end-to-end framework to identify fake news during natural disasters," in *Proc. Int. Conf. Inf. Syst. Crisis Response Manage.*, 2020, pp. 90–99.
[38] K. Pelrine, J. Danovitch, and R. Rabbany, "The surprising performance of simple baselines for misinformation detection," in *Proc. Web Conf.*, 2021, pp. 3432–3441.
[39] Y. Liu et al., "RoBERTa: A robustly optimized bert pretraining approach," 2019, *arXiv:1907.11692*.
[40] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," in *Proc. Int. Conf. Learn. Representations*, 2020, pp. 1–17.
[41] L. Cui and D. Lee, "CoAID: COVID-19 healthcare misinformation dataset," 2020, *arXiv:2006.00885*.
[42] K. Hunt, B. Wang, and J. Zhuang, "Misinformation debunking and cross-platform information sharing through twitter during hurricanes harvey and irma: A case study on shelters and id checks," *Natural Hazards*, vol. 103, pp. 861–883, 2020.
[43] A. M. Forati and R. Ghose, "Geospatial analysis of misinformation in covid-19 related tweets," *Appl. Geogr.*, vol. 133, 2021, Art. no. 102473.
[44] C. Bono, B. Pernici, J. L. Fernandez-Marquez, A. R. Shankar, M. O. Mülâyim, and E. Nemni, "Triggercit: Early flood alerting using twitter and geolocation - a comparison with alternative sources," in *Proc. 19th Int. Conf. Inf. Syst. Crisis Response Manage.*, 2022, pp. 674–686.
[45] C.-O. Truică, F. Rădulescu, and A. Boicea, "Comparing different term weighting schemas for topic modeling," in *Proc. IEEE Int. Symp. Symbolic Numeric Algorithms Sci. Comput.*, 2016, pp. 307–310.
[46] C.-O. Truică, E.-S. Apostol, and C. A. Leordeanu, "Topic modeling using contextual cues," in *Proc. 19th Int. Symp. Symbolic Numeric Algorithms Sci. Comput.*, 2017, pp. 203–210.
[47] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Int. Conf. Knowl. Discov. Data Mining*, 1996, pp. 226–231.
[48] C.-O. Truică and E.-S. Apostol, "TLATR: Automatic topic labeling using automatic (domain-specific) term recognition," *IEEE Access*, vol. 9, pp. 76624–76641, 2021.
[49] C.-O. Truică, E.-S. Apostol, J. Darmont, and I. Assent, "TextBenDS: A generic textual data benchmark for distributed systems," *Inf. Syst. Front.*, vol. 23, no. 1, pp. 81–100, 2020.
[50] C.-O. Truică, E.-S. Apostol, J. Darmont, and T. B. Pedersen, "The forgotten document-oriented database management systems: An overview and benchmark of native XML DODBMSes in comparison with JSON DODBMSes," *Big Data Res.*, vol. 25, 2021, Art. no. 100205.
[51] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.
[52] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[53] L. S. Snyder, M. Karimzadeh, R. Chen, and D. S. Ebert, "City-level geolocation of tweets for real-time visual analytics," in *Proc. ACM SIGSPATIAL Int. Workshop AI Geographic Knowl. Discov.*, 2019, pp. 85–88.
[54] M. Wick et al., "Geonames," 2015. [Online]. Available: http://www.geonames.org/
[55] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/
[56] T. Wolf et al., "Transformers: State-of-the-art natural language processing," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2020, pp. 38–45.
[57] L. AlSumait, D. Barbará, and C. Domeniconi, "On-line LDA: Adaptive topic models for mining text streams with applications to topic detection and tracking," in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 3–12.
[58] R. Řehuřek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proc. Workshop New Challenges NLP Frameworks*, 2010, pp. 45–50.
[59] C. Artho et al., "Model-based testing of apache ZooKeeper: Fundamental API usage and watchers," *Softw. Testing, Verification Rel.*, vol. 30, no. 7-8, 2019, Art. no. e1720.
[60] H. Jiang, "Density level set estimation on manifolds with DBSCAN," in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2017, pp. 1684–1693.
[61] D. Newman, J. H. Lau, K. Grieser, and T. Baldwin, "Automatic evaluation of topic coherence," in *Proc. Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2010, pp. 100–108.
[62] P. Gupta, Y. Chaudhary, T. Runkler, and H. Schuetze, "Neural topic modeling with continual lifelong learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3907–3917.
[63] W. Lukasiewicz, A. Todor, and A. Paschke, "Human perception of enriched topic models," in *Business Information Systems*. Berlin, Germany: Springer, 2018, pp. 15–29.
[64] M. Röder, A. Both, and A. Hinneburg, "Exploring the space of topic coherence measures," in *Proc. ACM Int. Conf. Web Search Data*, 2015, pp. 399–408.
[65] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI- 1, no. 2, pp. 224–227, Apr. 1979.
[66] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Commun. Statist.*, vol. 3, no. 1, pp. 1–27, 1974.
[67] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, 1987.
[68] M. Cheng et al., "A covid-19 rumor dataset," *Front. Psychol.*, vol. 12, 2021, Art. no. 1566.
[69] W. Y. Wang, ""Liar, liar pants on fire": A new benchmark dataset for fake news detection," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 422–426.
[70] H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi, "Truth of varying shades: Analyzing language in fake news and political fact-checking," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2931–2937.
[71] C.-O. Truică, N.-O. Istrate, and E.-S. Apostol, "A distributed automatic domain-specific multi-word term recognition architecture using spark ecosystem," in *Proc. 22nd Int. Symp. Parallel Distrib. Comput.*, 2023, pp. 31–38.
[72] Y. Q. Jinwen Xu, "Analysing information diffusion in natural hazards using retweets - a case study of 2018 winter storm diego," *Ann. GIS*, vol. 28, pp. 213–227, 2022.
[73] C.-O. Truică, E.-S. Apostol, M.-L. Erban, and A. Paschke, "Topic-based document-level sentiment analysis using contextual cues," *Mathematics*, vol. 9, no. 21, pp. 1–23, 2021.
[74] E.-S. Apostol, A.-G. Pisică, and C.-O. Truică, "ATESA-BÆRT: A heterogeneous ensemble learning model for aspect-based sentiment analysis," 2023, *arXiv:2307.15920*.
[75] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, 2013, pp. 160–172.
[76] I.-M. Rădulescu, A. Boicea, C.-O. Truică, E.-S. Apostol, M. Mocanu, and F. Rădulescu, "DenLAC: Density levels aggregation clustering–a flexible clustering method," in *Proc. Int. Conf. Comput. Sci.*, 2021, pp. 316–329.
[77] E.-S. Apostol, Ö. Coban, and C.-O. Truică, "Contain: A community-based algorithm for network immunization," *Eng. Sci. Technol., Int. J.*, vol. 55, pp. 1–10, 2024.
[78] C.-O. Truică, E.-S. Apostol, R.-C. Nicolescu, and P. Karras, "MCWDST: A minimum-cost weighted directed spanning tree algorithm for real-time fake news mitigation in social media," *IEEE Access*, vol. 11, pp. 125861–125873, 2023.

Elena-Simona Apostol received the PhD degree from the University Politehnica of Bucharest, Romania in 2014. She is an associate professor of Computer Science position with the Computer Science and Engineering Department, Faculty of Automatic Control and Computers, National University of Science and Technology Politehnica Bucharest, Romania. She was a postdoctoral researcher with Microsoft Research Center in Paris in collaboration with INRIA (The French Institute for Research in Computer Science and Automation) where she worked on state-of-the-art Big Data Analysis, Multi-Site Cloud Computing, and Bioinformatics. She was an invited researcher during the PhD studies with INRIA Rennes, France, working within the joint research team between KerData at INRIA and University Politehnica of Bucharest on Big Data management and analytics. During the bachelor's and master's studies, she was an intern and junior research engineer with the Fraunhofer FOKUS Institute, Berlin, Germany where she worked on Computer Networking and Telecommunications with a focus on mobile and service-orientated architectures. In 2022, she held a temporary teaching and research position with the Department of Information Technology, Uppsala University, Sweden. Her research focuses on Big Data, data management, parallel and distributed algorithms, machine learning, and data science.

Ciprian-Octavian Truică received the BSc degree in computer science and electrical engineering from the University Politehnica of Bucharest, in 2011, the BSc degree in computer science and mathematics from the University of Bucharest, in 2013, the MSc Degree in computer science engineering and information technology from the University Politehnica of Bucharest, in 2013, and the PhD degree in data management and text mining from the University Politehnica of Bucharest, Romania, in 2018. He is currently an assistant professor of computer science with the Computer Science and Engineering Department, Faculty of Automatic Control and Computers, National University of Science and Technology Politehnica Bucharest. In 2022, he was a postdoctoral researcher with the Department of Information Technology, Uppsala University, Sweden, where he worked on disinformation detection. From 2019 to 2020, he was a postdoctoral researcher with Data-Intensive Systems Group, Department of Computer Science, Aarhus University, Aarhus, Denmark where he was with Big Data Analytics for Time Series. In 2015 and 2016, he was an invited researcher with the ERIC laboratory, Universit de Lyon, France, where he worked on data management, machine learning, and natural language processing. His research interests mainly include to Big Data, data management, machine learning, text mining, natural language processing, and time series analysis. His research was also funded by The Nordic Observatory for Digital Media and Information Disorder (NORDIS) one of the national hubs of the European Digital Media Observatory.

Adrian Paschke is head of the Corporate Semantic Web group (AG-CSW) with a chair on semantic data intelligence with the institute of computer science, department of mathematics and computer science, Freie Universität Berlin (FUB). He additionally is director of the Data Analytics Center (DANA) with Fraunhofer FOKUS, director of RuleML Inc. in Canada, and professorial member with the Einstein Center Digital Future (ECDF), the Dahlem Center for Machine Learning and Robotics (DCMLR), the Institut für Angewandte Informatik (InfAI) with University of Leipzig, and founder of the Berlin Semantic Web Meetup group. With more than 200 peer-reviewed scientific publications he has made substantial scientific contributions in the field of semantic AI research and is active in standardization of semantic technologies (e.g., OASIS LegalRuleML, RuleML, OMG API4KB, W3C Semantic Web - W3C Rule Interchange Format, W3C RDF Stream Processing, etc.). He also served as an expert for industry and several ministries and funding bodies, including the European Commission. He was organizer and chair of renowned conferences and workshops (e.g. DEBS, RuleML, BIS, SWAT4HCLS, ODBASE, ESWC, Reasoning Web, Semantics, edBPM) and was invited speaker for keynotes, tutorials, panels, and lectures.