An Extensible Event Extraction System With Cross-Media Event Resolution

Fabio Petroni*, Natraj Raman*, Timothy Nugent*, Armineh Nourbakhsh†,

Žarko Panić*, Sameena Shah†, Jochen L. Leidner*‡

Thomson Reuters, Research & Development
FirstName.LastName@thomsonreuters.com

ABSTRACT

The automatic extraction of breaking news events from natural language text is a valuable capability for decision support systems. Traditional systems tend to focus on extracting events from a single media source and often ignore cross-media references. Here, we describe a large-scale automated system for extracting natural disasters and critical events from both newswire text and social media. We outline a comprehensive architecture that can identify, categorize and summarize seven different event types - namely floods, storms, fires, armed conflict, terrorism, infrastructure breakdown, and labour unavailability. The system comprises fourteen modules and is equipped with a novel coreference mechanism, capable of linking events extracted from the two complementary data sources. Additionally, the system is easily extensible to accommodate new event types. Our experimental evaluation demonstrates the effectiveness of the system.

KEYWORDS

event extraction, first story detection, event coreference, information extraction, news analytics

ACM Reference Format:
Fabio Petroni[1], Natraj Raman[1], Timothy Nugent[1], Armineh Nourbakhsh[2], Žarko Panić[1], Sameena Shah[2], Jochen L. Leidner[1][3]. 2018. An Extensible Event Extraction System With Cross-Media Event Resolution. In *Proceedings of 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, United Kingdom, August 19–23, 2018 (KDD 2018),* 10 pages.
https://doi.org/10.1145/3219819.3219827

*30 South Colonnade, London E14 5EP, UK.
†Work conducted whilst the author was at Thomson Reuters.
‡University of Sheffield, Department of Computer Science, 211 Portobello, Sheffield S1 4DP, England, UK.

1 INTRODUCTION

The ability to accurately extract breaking news events in real time, and to do so automatically, enables better decision-making in a broad range of fields including finance, security, policy/governance, NGO planning and disaster coordination efforts. For example, a government may use the output of an event extraction system to make better decisions regarding political unrest in a region; a trader may use event extraction to gain insights into companies vulnerable to natural disasters; or an NGO might map disasters to optimize the allocation of aid workers to where they are needed most. The task of identifying the earliest report of an event is known as *First Story Detection (FSD)* and is considered a challenging task given that events are typically reported multiple times, making it hard to distinguish between different instances of the same event [3].

In addition to detecting that something has happened, we are additionally interested in extracting a set of properties of such breaking events. This task, often referred to as *event extraction*, involves extracting attributes like the "5W1H" (i.e., who, what, where, when, why, and how). Essentially, the main goal of an event extraction system is to create a structured representation of the event from unstructured text. For instance, consider the following portion of text:

> Mogadishu truck bomb: Somalia's worst terrorist attack. 16 October 2017 - At least 300 people were killed and hundreds seriously injured.

An event extraction system should be able to digest this text and create the following structured event representation:

Event Type	terrorist attack
Location	Mogadishu
Time	16 October 2017
Impact	300+ killed, hundreds injured

In this paper we aim to extract breaking events and their attributes from both news and Twitter, and to identify coreferring event representations, including those that originate from different media (i.e., news and Twitter).

There is a long history of extracting events from news [11, 14] and, more recently, social media [1, 5, 40, 41]. Twitter, in particular, has proven to be a major source of breaking news across topics [23]. In a study of news stories reported on social media [37], Twitter led mainstream news media in more than 20% of disaster-related stories. This uniquely positions Twitter as an event-detection platform, especially for unexpected disasters and accidents [38].

While Twitter often reports breaking news faster than traditional news articles [23], it is less reliable. In contrast, agency news articles are rigorously verified before publication by seeking confirmation from multiple independent sources. Furthermore, Twitter only provides limited information about the event, while news articles present a rich context that completes the semantic picture about the event. Linking tweets and news articles enable information aggregation across two complementary media sources. To date, however, these two media types have never been combined in a large-scale event extraction system. Combining them is challenging due to their difference in text length and reporting style.

In this paper, we present an online event extraction system for both news articles and tweets, trained to recognize different event types - namely floods, storms, fires, armed conflict, terrorism, infrastructure breakdown and labour unavailability. The architecture of our system is complex: it contains fourteen components, and it makes use of multiple machine learning modules that use unsupervised, supervised and hybrid methods. Our system is also extensible: additional event types can be added with minor modifications. We consider a streaming setting in which news articles and tweets are continuously consumed and analysed. The extracted breaking news events are reported in a publish/subscribe paradigm. To the best of our knowledge, our system is the first to extract breaking events from an aggregated news and social media feed and to perform *event coreference across these two media sources.* Our novel coreference resolution algorithm makes use of event specific information such as location and time, and features extracted from the text. Our experiments show significant improvement over baseline methods.

The remainder of this paper is structured as follows: Section 2 discusses related work; Section 3 describes the method and implemented system; Section 4 provides a quantitative and qualitative evaluation and Section 5 summarizes our findings and concludes with suggestions for future work.

2 RELATED WORK

News Event Extraction. A large body of literature on event extraction from news articles is associated with information extraction competitions such as MUC-7 [8], *ACE2005* [11] and TAC-KBP2015 [12]. The Message Understanding Contest (MUC) is a series of US government research evaluations in information extraction, which included event extraction. The last evaluation, MUC-7, ran in 1996 and aimed to extract entities, relations and events from 158,000 New York Times articles. The Automatic Content Extraction challenge (ACE, 2000-2008) is a competition aimed at improving the state of the art in the extraction of named entities, relations and events. A multitude of solutions have been suggested to solve the event extraction task on the *ACE2005* corpus, ranging from feature-based methods [2, 18, 21, 22, 24, 27, 45] to neural approaches [7, 33, 34]. The formulation of the event extraction task in this paper is quite different from the one of the aforementioned competitions. In fact, while for the MUC and ACE contests the focus is on extracting all the event mentions in a news article, we are instead exclusively interested in breaking events and we assume that each news article contains at most one breaking event, as this is dictated by journalistic guidelines. Furthermore, we explicitly consider the inverted-pyramid style of writing news articles, where the most important parts of the story are placed in the beginning, as in [39, 43]. These latter works describe a multilingual event extraction system developed at the Joint Research Centre of the European Commission[1]. We differ in that we additionally monitor social media for events and we perform cross-media coreference. Other examples of event extraction systems from news include EMBERS [42], focused on civil unrest events, and EMBERS [4], where lexico-syntactic patterns are employed in order to discover events in the finance and politics domains, EventMiner [17] and Giveme5W [19].

Tweet Event Extraction. Several studies have attempted to implement real-time event detection from streaming tweets. The two prominent models are burst detection, and first story detection (FSD) [5]. Burst detection implements mechanisms that are sensitive to changes of volume in Twitter conversations. In contrast, FSD is designed for detecting stories as soon as they surface on Twitter, before the conversation around them reaches critical mass. As a result, FSD techniques are ideal for real-time detection of events when timeliness is a priority [28]. [40] present TwiCal, a system for open domain event extraction from Twitter (using a combination of a tagger specifically trained with Twitter data, the TempEx time tagger and LinkLDA), i.e. they do not posit a pre-defined, closed inventory of events. Unlike our system, theirs only operates on social media, and does not address coreference across media.

Cross-Media Event Coreference. State of the art solutions to link tweets to news articles are mainly based on unsupervised approaches. A popular application for such linking exercise is the summarization of news articles driven by social information [32, 44]. Closer to our objective is the work of Guo et al. [16]. They apply a latent variable model, namely the Weighted Textual Matrix Factorization (WTMF) [15] to both the tweets and the news articles text in order to assess how similar they are. The similarity is measured by the cosine similarity of the latent vectors associated with the tweet and the news article. However, this solution is not suitable for an online setting like the one we consider in this paper, where news and tweets continuously flow into the system, since WTMF assumes a static and predefined set of news and tweets. Moreover, current solutions aim at linking together generic news article and tweets while we exclusively focus on coreferencing news and tweets that refer to the same event. We do so by using a novel supervised algorithm that takes into account event specific attributes and work in a streaming setting.

3 SYSTEM DESCRIPTION

Figure 1 presents an overview of the system architecture. The input *News Feed* and *Twitter Feed* are processed continuously by a *News Event Extraction Pipeline* and a *Tweet Event Extraction Pipeline* respectively. These two pipelines produce *News Event Representations* and *Tweets Event Representations* that are processed by a *Coreference Resolution* component, that enriches these event representations with links to other representations for the same event and generates an *Event Feed.* An *Elasticsearch*[2] database instance is used to persist and retrieve event representations. The functions of the two

[1]http://emm.newsbrief.eu/NewsBrief/clusteredition/en/latest.html
[2]https://www.elastic.co

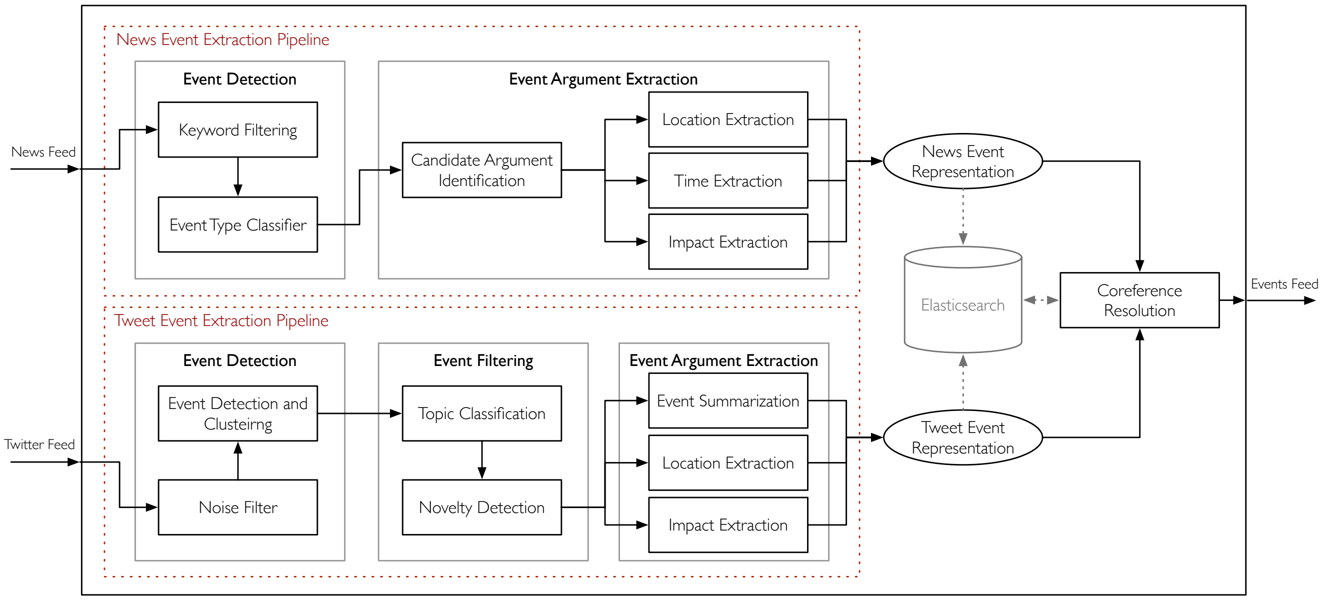


Figure 1: System architecture overview

pipelines and the coreference resolution component are elaborated in the following subsections.

3.1 News Event Extraction

A large-scale commercial news feed, proprietary to Thomson Reuters, which aggregates news content from over 12,000 sources is used as input to the news event extraction pipeline. The pipeline performs two key functions: (1) detecting if a news article reports a breaking event and if so, (2) extracting the event arguments such as location, time and impact from the article text.

Event Detection. First we employ a pre-filtering step in which a list of keywords is used to exclude articles that do not contain at least one event-related term in the article text. This allows the system to handle a large volume of news articles.

Second, all news articles that pass the pre-filtering step are analysed by a supervised classifier that discards or assigns an event type to them. We assume that each new article contains at most one event and framed the problem as a multiclass supervised learning task, where the objective is to train a classifier capable of accurately predicting the discrete class label y_i, where $y_i \in$ {'conflict', 'fire', 'flood', 'infrastructure breakdown', 'labour unavailability', 'storms', 'terrorism', 'none'}, for a given news article x_i. In previous work we compared different classification approaches for the task, namely Support Vector Machines (SVM), Random Forest, Convolution Neural Network and Hierarchical Attention Network [36]. We showed that a custom SVM-based approach, which explicitly separates the information contained in the title and in the body of the news article, outperforms all other approaches, achieving a weighted F1-score of 77.3% when evaluated on a corpus of $2.5k$ human-annotated agency news articles.

We used word embeddings as input features in all classifiers. In particular, we trained a 200-dimensional fastText [6] model using a combination of two data sources. As well as the filtered English Wikipedia dump from 2006, we included tokens extracted from news articles tagged with disaster or accident topic codes, allowing the model to capture the semantic structure of event-related news. In previous work [36], we empirically showed that this approach leads to improvements compared to the use of publicly available pre-trained embedding vectors, since our vectors are able to capture event-specific syntactic and semantic information that are beneficial for the classifier. In the rest of the paper, unless otherwise specified, we refer to this fastText model when we mention word embedding vectors.

Candidate Arguments Identification. We used NLP4J [9, 10] to perform linguistic processing on the news article text. The toolkit's tokenizer splits the raw text into tokens by their morphological aspects. It also provides additional information about a token such as its part-of-speech tag, named entity type and a dependency tree. These enriched token attributes are leveraged in the further stages of the pipeline to compute informative features. For example, the part-of-speech tags capture the syntactic structure around the words while the dependency trees can resolve structural ambiguity. Recognizing the mentions of entity types such as locations, dates, numerals etc. in the text, provides a set of candidate event arguments. This efficiently narrows down the search space for extracting the true event arguments such as the event location, event time and event impact.

Location Argument Extraction. We define a geographical taxonomy where locations are split according to a four-level hierarchy:
(level 0) country;
(level 1) first administrative area (e.g., state, province, etc.);
(level 2) second administrative area (e.g. county, department, etc.);
(level 3) localities (e.g. city, towns, villages, etc.).
Using this taxonomy each location can be represented as a tree.

Our goal here is to extract the event location among all the candidate locations. In fact, a news article might contain locations that are not related to the event (e.g., the location where the news article was published). Moreover, we are interested in the most specific location in case multiple locations are extracted. For instance, in the example in the introduction, the most specific location “Mogadishu” is extracted as the event location while “Somalia” is discarded. To solve this task we used a supervised classification approach. In particular, for every candidate location identified by the NLP toolkit, a real-valued feature vector is obtained using the concatenation of the following vectors:

(1) The average of the word embedding vectors corresponding to the sentence in which the candidate location is present.
(2) A binary vector whose dimensions correspond to the possible entity types. The value at each dimension is set to 1 only if the entity type appears in the surrounding k tokens of the candidate location (we used $k = 5$).
(3) A binary vector whose dimensions correspond to the possible part-of-speech tags. The value at each dimension is set similar to above.
(4) The position token offset of the candidate location in the news article.
(5) A 4-dimensional binary vector that encodes the location representation in the four-level hierarchy.
(6) A binary variable which is set to 1 if the article contains another location that is more specific (i.e., has a higher lever in the taxonomy) than the candidate location.

The last two components of the feature vector encode the geographical taxonomy while the rest of the features capture the syntactic and semantic context. Given training set pairs of the above feature vectors corresponding to the candidate locations and binary variables indicating whether a candidate location is an event location, an SVM classifier is trained. During prediction, each candidate location is tested with the trained SVM model. In case the classifier generates multiple positive predictions, the candidate location for which the SVM model outputs the highest confidence score is selected.

Our system also determines the geographical coordinates corresponding to the event location. A problem to address is the location ambiguity: several distinct locations may have the same name. For example, if the event location is identified as “Naples", it is important to disambiguate whether the location referred is “Naples, Italy" or “Naples, Florida (US)" or “Naples, Illinois (US)". The ambiguity in the event location is resolved here based on spatial proximity clues. It is assumed that all the candidate locations are likely to be near to each other (hence to the event location). When the event location is ambiguous, we compare all potential addresses with the remaining canditate locations to disambiguate it. In detail, a geocoder[3] is queried to retrieve all the potential addresses corresponding to all candidate locations. Each address is arranged in the four-level hierarchy described above. For each potential address of the event location we compute a score, that is the linear combination of an *overlap score* and a *popularity score*. The overlap score is computed by summing the height of the common subtrees between the potential event location address and all other candidate locations addresses. The popularity score is returned by the geocoder and is calculated using frequency-based statistics over Wikipedia articles. Finally, the address with the maximum score is selected and the corresponding geographical coordinates are used.

[3]https://nominatim.openstreetmap.org.

Time Argument Extraction. A rule-based model is employed to determine which of the candidate temporal expression is the event date/time argument. The following four types of temporal expressions are considered here: absolute values (e.g. *12-March*), explicit offsets (e.g. *yesterday*), implicit offsets (e.g. *Thursday*) and positional offsets (e.g. *last week*). We consider the event date/time to be the first occurring temporal expression in the article text. An exception to this rule is when the following two conditions are simultaneously true: (1) the news article began with an absolute value (usually the publication date/time) and (2) the first sentence of the news article contains multiple temporal expressions. In this case we ignore the first absolute value and consider the second temporal expression. The selected temporal expression is converted to a canonical form, with the publication timestamp of the article used to resolve offsets (e.g, *yesterday, last week*).

Impact Argument Extraction. In addition to location and time, we also attempted to extract a measure of the event's human impact. Typically, the human impact of large-scale events is expressed in quantifiable units in association with an effect (e.g. *ten people injured*, *15 drowned*). To leverage this, we applied our impact extraction function to sentences that contained tokens with a cardinal number part-of-speech tag. We considered this value as the putative unit of human impact (*ten, 15*), and proceeded to search in the vicinity of this value for word sequences that were likely to describe the effect (*injured, drowned*). Using a dataset of annotated examples, we applied a supervised machine learning model to this task. In particular, we used an SVM classifier to identify sequences that, when paired with the numeric value, did or did not indicate human impact. Putative effect word sequences were generated by constructing all n-grams, where $1 \leq n < 5$, from each side of the cardinal token within the sentence. For each word sequence, the following features were encoded into a vector:

(1) the average embedding vector corresponding to the word sequence;
(2) the length of the word sequence (n);
(3) the pre and post token offset of the cardinal number token, relative to the word sequence;
(4) a binary vector corresponding to one part-of-speech tag for each word in the word sequence;
(5) a binary vector corresponding to the entity types of the word sequence;
(6) a binary vector corresponding to the dependency tree relations of the word sequence.

Note that the numeric value itself was not included as a feature. When applied to unseen news articles, the SVM model returns predicted human impact as pairs of (amount, effect). The extracted effects are then mapped into four broad categories (i.e., dead, injured, missing and displaced) based on a dictionary of keywords.

The result of the event extraction pipeline for news articles is a live stream or feed of *News Event Representations* containing the event type, location, time and impact of an event. These representations are persisted into an Elasticsearch database instance and passed as input to the coreference resolution component.

3.2 Tweet Event Extraction

Detecting newsworthy events from Twitter poses different challenges from event extraction from news media. Tweets can discuss a variety of topics, many of them unrelated to events. It is important to identify and filter unrelated messages such as spam, opinion, and general chitchat, before any attempt is made to extract events from tweets. Once non-news-related tweets are filtered, our system clusters the remaining tweets around the events that they discuss. Each event is then enriched with metadata such as location and impact. In the following, we describe the event extraction pipeline from Twitter. For some components we provide just an overview; a detailed description can be found in previous works as cited below.

Noise Filtering. In our definition any message that does not convey a newsworthy event is considered *noise*. This surpasses the traditional definition of *spam* as advertisements and bot-generated content, and includes daily chit-chat. By this definition, in the Twitter universe the signal-to-noise ratio is very low, since a tiny minority of tweets discuss news events (about 0.2%). We filter noise by applying an iterative filter to the tweets. First, a rule-based classifier is used to filter suspicious spam users or messages from certain domains such as `ebay.com`. Next, we use a topic model to identify chit-chat. The model is trained on two corpora of online conversations that are unrelated to news. Finally, we apply cost-sensitive learning [20] to the remaining messages. Since the signal-to-noise ratio is very low, the model is tuned to penalize false positives such that messages that may have some valuable content in them are not filtered [28].

Newsworthy Event Detection and Clustering. The outcome of the previous step is a stream of tweets that have passed all noise filters. Next, this stream is inspected for potential event-related content. We conceptualize an event as a semantic entity with four main dimensions: *what*, *where*, *who*, and *when*. We use OpenCalais[4] to identify the first three dimensions (if present) in each tweet. For the fourth dimension, we use a rule-based model that identifies explicit or implicit expressions of time such as "on Monday," "this morning," or "1926." A soft-matching process is used to align tweets along each dimension, and a linear classifier is trained on the interpolation of these dimensions to group tweets around real-world events [25]. The result is a cache of clusters, where each cluster consists of tweets that discuss a particular event. This model is a *First-Story Detection* (FSD) algorithm [5] that can be used to identify events dynamically and in real-time: if a cluster forms around an event, as new tweets are posted about the same event, they can be dynamically added to the cluster. If a cluster remains idle for more than 24 hours, it is removed from the live cache.

Topic Classification. The events detected so far are not guaranteed to be newsworthy in any capacity. Tweets can discuss events such as birthdays, happy hours and concerts. To filter non-newsworthy events, we use a topic classification method that is modeled on tweet content [26]. First, we train a LDA model on a corpus of 500 million tweets, producing 300 topic distributions. Representing each tweet by the concatenation of two one-hot encoded vectors (one representing the word's index in the dictionary and another representing its topic), we train a skip-gram model [31] to jointly predict the word as well as its topic. The resulting 600-dimensional embeddings are used as features for a SVM model, which is trained on a set of 26,300 tweet labeled by the topics introduced in section 3.1. A new, catch-all topic is introduced to capture tweets that do not fall under any of the target topics, which are then removed from the dataset.

Novelty Detection. Tweets that discuss event-related topics do not necessarily discuss them in the context of current events. Wars, geopolitical events and natural disasters that have taken place days or even years ago often resurface on Twitter conversations. To ensure that only current events are captured, we assign a novelty score to each tweet cluster. The score is calculated using a hybrid approach [25]. First, tweets that explicitly mention a historical event such as WWII or 9/11 are removed by a taxonomy-based filter. Next, tweets that mention an expression of time such as "last week" are removed by the same filter that is used for event extraction. Finally, for every new tweet cluster that is formed, a pairwise similarity score is calculated between that cluster and every cluster in the cache. If the incoming cluster closely resembles an older one, it is likely to be an update on an event that is previously reported (e.g. an update on a natural disaster that confirms the number of casualties). These residual updates are ignored and removed.

Event Summarization. In order to generate event representations from tweets that line up with the event representation extracted from news articles, each tweet cluster has to be represented by a summary that functions as its "headline." The summary is generated by selecting one of the tweets in the cluster that is most representative, objective, and informative. Given a cluster, we treat each tweet as a document and represent it by a tf-idf vector. Each cluster is then represented by a centroid vector. We score each tweet vector based on its similarity to the centroid. Using a rule-based approach, we penalize tweets that include opinionated terms or patterns such as repeated characters or punctation. The tweet with the highest score is selected as the summary [29].

Location Extraction. We use a Twitter-specific geo-parsing method to handle Twitter lingo such as hashtags and mis-capitalized words [13]. The resulting toponyms are checked against the *Nominatim*[5] gazetteer. Within each cluster, a least-common-distance metric is used to disambiguate toponyms such as "Paris" that can be mapped to multiple coordinates around the word.

Since tweets have a strict character limit, the number of locations mentioned in them is often limited. More than 60% of tweets in our dataset mention a single location, and fewer than 2% have more than three locations in them. As a result, we use a simple approach to extracting the primary location of each event. If multiple locations are mentioned but some are included within others, the less granular locations are ignored. If the remaining pool of locations includes more than one location, the location mentioned last in the tweet is selected. This is based on the observation that in most cases, the

[4]http://www.opencalais.com

[5]https://nominatim.openstreetmap.org.

last location mentioned in the tweet is the primary location of the event (e.g. in "Six Islamic State leaders escaping Hawija killed in Diyala."). Using this approach, each tweet generates one location for the cluster. A voting system is used to select the primary location for the entire cluster.

Time and Impact Detection. The tweet's timestamp is treated as the event temporal expression. We do not detect temporal expressions in them since: (1) tweets stream in real-time and (2) the novelty detection algorithm makes sure they are discussing current events. Moreover, we associate with each cluster a temporal expression that is the timestamp of its formation. Finally, we add an annotation of impact to each cluster, following the same approach we used for news articles (described in section 3.1).

The result of the extraction pipeline for tweets is a live stream of filtered and annotated tweet clusters that represent current events. These *Tweet Event Representations* are persisted into the Elasticsearch database instance and passed as input to the coreference resolution component.

3.3 Coreference Resolution

The coreference resolution component takes in input two event representations, r_x and r_y, and output a boolean value that indicates if the input representations refer to the same event. This linking procedure is achieved using a novel supervised classification approach. We first introduce the similarity scores used as features by the classifier.

Spatial Similarity. We consider all extracted locations to compute the spatial similarity score between event representations. Let $X = [x_1, x_2, ..., x_n]$ be the vector of n locations extracted from r_x and $Y = [y_1, y_2, ..., y_m]$ be the vector of m locations extracted from r_y. We use the geographical taxonomy described in Section 3.1, where a location is split into a four-level hierarchy. Using this taxonomy each location can be represented as a tree. Similarity between two locations x and y is based on the length of the common path, defined as:

$$\mu(x, y) = \frac{\lambda(x \cap y)}{\lambda(x) + \lambda(y)} \tag{1}$$

where $x \cap y$ is the maximum common subtree between x and y and $\lambda(x)$ is the height of tree x. Following [30], we define the spatial similarity between two locations vectors X and Y as:

$$S_L(X, Y) = \frac{\sum_{i=1}^{n} \max(\{\mu(x_i, y_j) | \forall y_j \in Y\}) + \sum_{j=1}^{m} \max(\{\mu(x_i, y_j) | \forall x_i \in X\})}{m + n} \tag{2}$$

Temporal Similarity. We consider all extracted temporal expressions to compute the temporal similarity score between event representations. Let $T = [t_1, t_2, ..., t_z]$ be the vector of z temporal expressions extracted from r_x and $V = [v_1, v_2, ..., v_w]$ be the vector of w temporal expressions extracted from r_y. We define the temporal similarity between two temporal vectors T and V as the minimum absolute time interval between temoral expressions in vector T and in vector V, as follows:

$$S_T(T, V) = min(\{| \ t_i - v_j \ | \ | \forall t_i \in T, v_j \in V\}) \tag{3}$$

Entity Similarity. We consider all the extracted named entities to compute the entities similarity score between event representations. In particular, we use the Jaccard similarity coefficient between the sets of persons (i.e., S_P) and organizations (i.e., S_O).

Text Similarity. We consider the text of the news article or the summarization of a tweet cluster to compute the text similarity score between event representations. In particular, we represent the text of r_x and r_y as a vector, $r_x^{\vec{txt}}$ and $r_y^{\vec{txt}}$, using the following procedure: given a tokenized text $A = \{a_1, ..., a_q\}$ and given a pre-trained word embedding model that maps each token a_i to a word vector $\vec{w_i}$, the vector representing the text is computed as:

$$r^{\vec{txt}} = \frac{\sum_A \vec{w_i}}{|A|} \tag{4}$$

The cosine similarity between $r_x^{\vec{txt}}$ and $r_y^{\vec{txt}}$ is then used as text similarity feature. Following a previous study [36], we actually define two distinct text similarity features, one considering only the title of news article (i.e., S_E) and one considering only the body (i.e., S_B). For tweets this distinctions is clearly unfeasible and, in case of cross-media coreference, S_E is defined as the cosine similarity between the news article title vector and the tweet text vector while S_B as the cosine similarity between the news article body vector and the tweet text vector.

Online Coreference Model. We trained a binary SVM classifier to detect if two event representations are coreferent or not, using as features all the aforementioned similarity measures. In particular, we build the following 6-dimensional feature vector for the pair of event representations (r_x, r_y):

$$f_{r_x, r_y} = \{S_L, S_T, S_O, S_P, S_E, S_B\} \tag{5}$$

The trained coreference model is applied online by comparing the current event representation with all representations retrieved from the Elasticsearch database instance in a configurable time window (we used 72-hours). Note that a major short-coming of previous news-tweets coreference methods is that they require access to all data at training time, and hence are not applicable in an online setting. Instead, our method for establishing event coreference, once trained offline, is able to operate with unseen event representations pairs in an online fashion, and works well with a small time window of past tweets and news articles leading up to the present. This property is crucial for a near real-time system.

Finally, event representations, enriched with links to other coreferent representations, are published using a Kafka queue. This produces an industrial scale live events feed that can be subscribed to by clients.

3.4 Extensibility

In order to extend the system with an additional event type (e.g., *drought*) a small number of actions are required. For the news extraction pipeline the following steps are necessary:

(1) definition of a list of trigger keywords for the new event type (e.g., "water scarcity", "famine").
(2) A set of annotated news articles (e.g., a few hundreds) with positive and negative instances for the new event type.

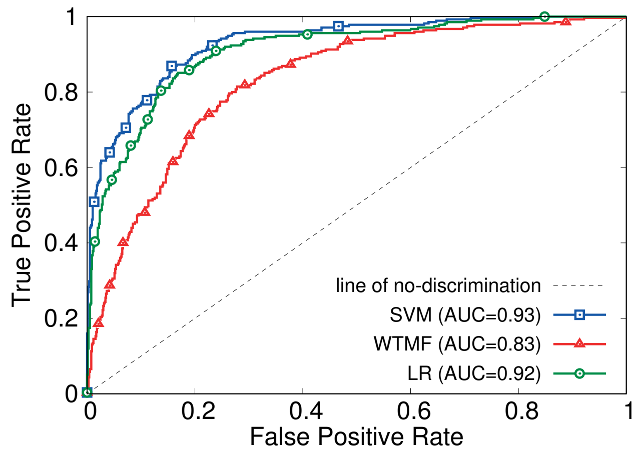


Figure 2: Coreference classification-based experiments results - ROC curve.

Table 1: Coreference ranking-based experiments results .

algorithm	$ATOP_{100}$	$TOP(10)$	RR
WTMF	0.914	0.815	0.516
LR	0.929	0.865	0.596
SVM	0.964	0.943	0.651

(3) Retrain the event event type classifier including the new event type.

Similarly, adding a new event type in the Twitter pipeline requires modification only in the topic classification algorithm. In particular, a dataset annotated with new topics reflecting the new event type must be created and the SVM classifier will have to be retrained on it. The above steps can be done with a reasonable amount of effort. No changes are needed in the algorithms to extract event arguments or in the coreference resolution component. Also the word embedding features do not need to change, since they are learned in an unsupervised fashion.

4 EVALUATION

This section reports the results of experimental studies we conducted on internal components of our event extraction system. Moreover, we report some statistics for a live instance of our system as well as some real examples of extracted events.

4.1 Linking Tweets to News Articles

In this section we evaluated the performance of our SVM-based coreference resolution algorithm in the task of linking tweet and news articles referring to the same event.

Data and Methodology. To create a corpora of tweets/news pair referring to the same event we adopted the following strategy: (1) harvest a collection of ~ 25000 tweets extracted by our system (see Section 3.2) that contain a URL link in their body; (2) navigate each URL and, if the link refers to a news article, retrieve title and body ; (3) ingest these information in the system pipeline for news (see Section 3.1) (4) if the news article is identified as containing an event consider the article and the original tweet as linked - they refer to the same event. We were able to form 5496 tweets/news pairs with this strategy. Moreover, in order to provide the classifiers with negative evidence we randomly sampled a collection of tweets/news pairs (that are not supposed to refer to the same event) by arbitrarily linking together tweets and news articles from the positive pool that don't appear linked in a positive pair. We collected as many negative examples as the number of positive pairs.

We used 80% of this data to train our coreference algorithm (SVM) and two competitors: (1) a Logistic Regression (LR) classifier that uses in input the same feature as our SVM-based approach; (2) the Weighted Textual Matrix Factorization (WTMF) of [15, 16]. All models have been tuned with a fine-grained grid search over the space of the hyperparameters. We used the remaining 20% of the data, filtering out "trivial" cases where the tweet content is simply the news title, as test set and performed a ranking-based set of experiments that closely follows the evaluation in [16]. In particular, for each (T, N) (tweets/news) pair in the test set we created a news article pool by retrieving from our Elasticsearch instance: (i) the 100 news articles with the closest publication time to the article N publication time, published after or before N, and (ii) the 100 news articles published before the tweet T with the closest publication time to the tweet T publication time. Ideally, a good algorithm should be able to identify all the news articles (if more than one) in the pool that refer to the same event of the one described in the tweet T, and additionally should output an higher confidence for the news article specifically referred to by the URL in the tweet.

Therefore, following Guo et al. [16] we used three ranking metrics to evaluate the performance of competing solutions:

- Reciprocal Rank (RR) = 1/rank. For instance, RR=1/3 when the correct news article is ranked in 3rd highest place.
- $TOP(10)$ = 1 if the URL referred news article is in the "top 10" list, 0 otherwise.
- Area under the TOP(k) recall curve ($ATOP_{100}$) = $1/100 \cdot \sum_{k=0}^{100} TOP(k)$.

The ranking is given by the cosine similarity between the news article and tweet vectors for WTMF, while for LR and SVM is given by the confidence score of the classifier for the tweets/news pair.

We additionally manually annotated a set of 1,138 tweets/news pairs as coreferent or not (370 positive cases and 768 negative) and we performed a classification-based evaluation campaign. Note that, while LR and SVM are able to output a classification prediction (i.e., tweets/news coreferent or not), WTMF only outputs a real value representing the similarity between the tweet and the news article in input, without providing any sort of discrimination threshold. To allow a fair comparison we used two evaluation metrics that don't require a predefined discrimination threshold, namely the receiver operating characteristic curve (ROC curve) and the area under the ROC curve (AUC).

Results. Table 1 reports the results of our ranking-based set of experiments. Our SVM-based solution outperforms competitor solutions by a large margin. It ranks the correct pair in the TOP-10 more than 94% of the time, with respect to the 87% of LR and 82% of WTMF. Also the values of $ATOP_{100}$ and RR are in favour

Table 2: Location extraction accuracy.

algorithm	strict	lenient
baseline	0.52	0.60
LR	0.57	0.71
RF	0.53	0.70
SVM	0.58	0.72

Table 3: Impact extraction results.

algorithm	precision	recall	F1-score
baseline	0.83	0.74	0.72
LR	0.89	0.89	0.89
RF	0.89	0.88	0.88
SVM	0.91	0.90	0.90

of SVM. The SVM is also the best performing algorithm for the classification-based experiments, reported in figure 2. Here the ROC curve visually illustrates the performance of the competing solutions by plotting True Positive Rates (TPR) values against False Positive Rates (FPR) values obtained by varying the discrimination threshold of the binary classifier. The diagonal line of this graph (the so-called line of no-discrimination) represents a completely random guess. A perfect classifier would be represented by a constant line with TPR=1 for all FPR values, with a resulting AUC of 1. Our SVM-based algorithm achieves an AUC of 0.93, outperforming both LR (0.92) and WTMF (0.83). These experiments clearly shows the importance of considering additional features such as location and time, together with text-based features to be able to correctly link tweets and news referring to the same event.

4.2 Location Extraction

We manually extracted the event location from 1,200 news articles that described breaking events. 70% of these articles were used as training set and the remaining 30% for evaluating location argument extraction methods. As baseline, we considered a solution that always picks the first candidate location in the text as event location. Moreover, we trained three different supervised classification algorithms namely Logistic Regression (LR), Random Forest (RF) and SVM using the features described in Section 3.1.

We evaluate the prediction accuracy in two ways: *strict match* and *lenient match*. In strict match, an extracted location is required to exactly match the manually annotated location while in lenient match it is sufficient if the extracted location overlaps with the geographic taxonomy of the manually annotated location. For instance, in the example presented in the introduction extracting the location "Somalia" is considered a correct prediction for the lenient match metric, since "Somalia" contains "Mogadishu", while is considered an incorrect prediction for the strict match metric.

The location extraction results are presented in Table 2. There is an improvement for both strict match accuracy and lenient match accuracy when using a supervised classification approach. Further, the SVM classifier outperforms the other classifiers. We also observed that in over 90% of the articles, the event location appears within the first 5 sentences of the text. This heuristic can be applied to reduce the number of candidate locations.

Table 4: Performance metrics for each component in the tweet event extraction pipeline.

Component	Model	Performance	Metric	Detail
Noise filtering	Cost-sensitive learning	78.8%	F1 score	[28]
Event detection	FSD + semantic term matching	78%	B-cubed	[25]
Topic classification	Topical word embeddings + SVM	84.4%	F1 score	[26]
Novelty detection	Rule-based +Linear interpolation	85.88%	F1 score	[25]
Event summarization	Vectorized distance	3.6/4	Human evaluation on 4-point scale	[29]
Location extraction	Hybrid	78.7%	F1 score	[13]
Impact detection	SGD	89.54%	F1 score	[35]

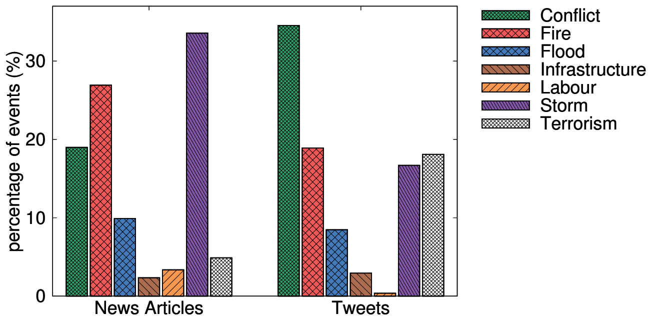


Figure 3: Statistics about events extracted from news articles (3,149) and tweets (3,163) for the month of September 2017.

4.3 Impact Extraction

We manually annotated the human impact of events in about 1,000 news articles and used 70% of these articles for training and 30% for evaluation. As baseline, we built a gazetteer of over 100 phrases such as *casualties*, *suffered from fractures*, *not accounted for* etc. and extracted the effect by checking whether the article text contained any of these phrases. Similar to location extraction, we trained three different supervised classification algorithms. Prior to training, we observed a substantial class imbalance between positive examples (i.e. the number of word sequences that described an effect) with respect to negative examples. Hence, we down-sampled the negative examples to balance the training data. The features described in Section 3.1 were used. The precision, recall and F1-Score for the various methods is presented in Table 3. The classification based methods outperform the baseline method substantially. The difference in performance between the various supervised classification methods is small, with the SVM producing the best results.

4.4 Twitter and System Statistics

We have already presented the performance metrics of the models used in the Twitter pipeline in previous works. Table 4 summarizes these results. Additional details of the experiments can be found in the paper cited in the 5th column.

event (news)

Event Type	fire
Location	northern California
Time	9 October 2017
Impact	-

Evacuations underway as fires rage through California's wine country.
October 09, 2017 - A hospital and homes had to be evacuated overnight in northern California after fast-moving wildfires...

↕

event (Twitter)

Event Type	fire
Location	Northern California
Time	9 October 2017, 10:07
Impact	-

@qatarupdate: Wildfires force evacuations in Northern California #news
@WISN12News: Wildfires force evacuations in Northern California

Event Type	storm
Location	Aglish
Time	16 October 2017
Impact	1 killed

Woman in her 20s killed in Hurricane Ophelia storms after tree falls onto her car in village.
A woman has been killed after a tree fell onto her car during fierce Hurricane Ophelia storms. The accident happened in the Irish village of Aglish...

↕

Event Type	storm
Location	Waterford
Time	16 October 2017, 12:53
Impact	1 dead

@lappinm: Ophelia: One dead after tree falls on car in Waterford via @RTENewsNow
@SkyNewsBreak: RTE reports one person has died in southeast Ireland as a direct of storm Ophelia

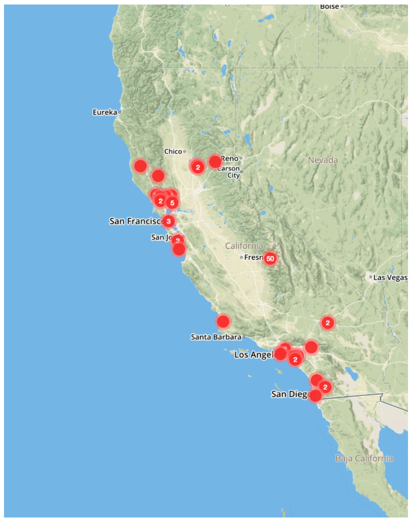
Event Type	conflict
Location	Baghlan
Time	07 October 2017, 15:09
Impact	9 killed, 1 injured

Militant attack kills 2 police officers in Afghanistan.
Unknown militants in Afghanistan have attacked a police checkpoint in the country's northern Baghlan Province, killing two officers and injuring another [...] seven Taliban militants had been killed...

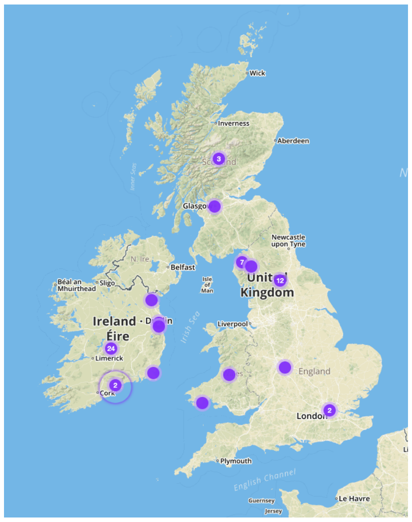
↕

Event Type	conflict
Location	Afghanistan
Time	08 October 2017 10:10
Impact	9 killed

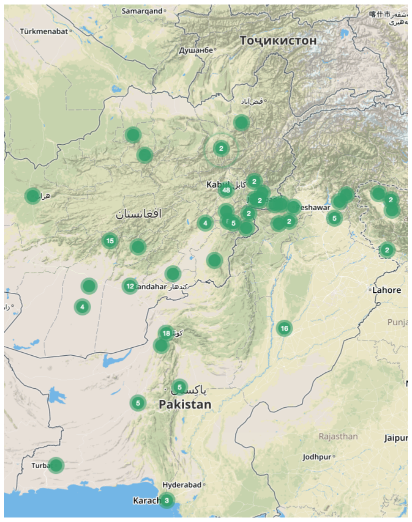
@1stNewsHeds: ABC News: Afghan officials: 2 police, 7 militants killed in attacks
@HighGroundVets: Afghan officials: 2 police, 7 militants killed in attacks - Nasratullah Jamshidi



(a) California's wildfires



(b) Hurricane Ophelia storms



(c) Conflicts in Afghanistan

Figure 4: Example of events extracted by our system. The top part of the figure shows an event representation extracted from a news article and from a tweet, respectively, which have been identified as referring to the same event by our coreference resolution algorithm (indicated by the arrow). The bottom part shows all events detected in the first three weeks of October 2017 for event type (a) fire, (b) storm and (c) conflict as points on a map.

Figure 3 reports some statistics about the events that have been extracted by our system in the month of September 2017. The left part of the figure shows the composition of the 3149 event representations extracted from news articles, while the right part of the figure shows the composition of the 3163 event representations extracted from tweets.

4.5 Sample Output

Figure 4 reports some examples of events that happened in the month of October 2017. In particular, three examples of event representations extracted from Twitter and news articles successfully linked together by our coreference resolution algorithm are presented, connected to the wildfires that affected California (Figure 4a), the Hurricane Ophelia storms in Ireland and United Kingdom (Figure 4b) and armed conflicts in Afghanistan (Figure 4c), respectively. Moreover, all the events extracted for these three event types (i.e., fire, storm and conflict) for the first three weeks of October 2017 in those areas are showed as points on a map, using the coordinates of the extracted event location.

5 CONCLUSIONS & FUTURE WORK

We have described a new system for cross-media event extraction and event coreference that can process agency news and micro-blog posts from Twitter. Our extensible system has been trained to identify, classify, and resolve 7 types of disaster events, and makes them available to downstream client applications via a publish-subscribe mechanism as an event stream. Our novel coreference model enriches this event stream with links between representations that stand for the same event, including those that bridge between Twitter and news, which reduces the total number of alerts.

In future work, the inventory of event types can be expanded. In particular, more finance-relevant events (e.g. mergers & acquisitions) could add value to the platform, and they have been shown to be viable in other contexts. The system could also be trained on data in languages other than English. Our semi-supervised learning methods could also be integrated to improve the development turnaround for new event types. The speculations about—and updates of—casualty numbers and analysing the evidence cited to support these numbers is an interesting field for further research. We also plan to apply the system retrospectively to a digital news archive to create a large events database.

ACKNOWLEDGMENTS

The authors would like to thank Khalid Al-Kofahi and Shaun Sibley for supporting this project and to the three anonymous KDD reviewers for their constructive feedback.

REFERENCES

[1] Puneet Agarwal, Rajgopal Vaithiyanathan, Saurabh Sharma, and Gautam Shroff. 2012. Catching the Long-Tail: Extracting Local News Events from Twitter.. In *Proceedings of The International AAAI Conference on Web and Social Media.*

[2] David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events.*

[3] James Allan, Victor Lavrenko, and Hubert Jin. 2000. First Story Detection in TDT is Hard. In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM).*

[4] Chinatsu Aone and Mila Ramos-Santacruz. 2000. REES: a large-scale relation and event extraction system. In *Proceedings of the sixth conference on Applied natural language processing.*

[5] Farzindar Atefeh and Wael Khreich. 2013. A Survey of Techniques for Event Detection in Twitter. *Computational Intelligence* (2013).

[6] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. In *arXiv preprint arXiv:1607.04606.*

[7] Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, Jun Zhao, et al. 2015. Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks.. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing.*

[8] Nancy A. Chinchor. [n. d.]. Overview of MUC-7. In *Proceedings of the Seventh Message Understanding Contest.*

[9] Jinho D Choi. 2016. Dynamic Feature Induction: The Last Gist to the State-of-the-Art.. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

[10] Jinho D Choi and Martha Palmer. 2012. Fast and robust part-of-speech tagging using dynamic model selection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics.*

[11] George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The Automatic Content Extraction (ACE) Program-Tasks, Data, and Evaluation.. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC).*

[12] Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie Strassel. 2015. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *Proceedings of TAC KBP 2015 Workshop, National Institute of Standards and Technology.*

[13] Judith Gelernter and Shilpa Balaji. 2013. An algorithm for local geoparsing of microtext. *GeoInformatica* (2013).

[14] Ralph Grishman. 1997. Information extraction: Techniques and challenges. In *Information extraction a multidisciplinary approach to an emerging information technology.* Springer.

[15] Weiwei Guo and Mona Diab. 2012. A simple unsupervised latent semantics based approach for sentence similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics.*

[16] Weiwei Guo, Hao Li, Heng Ji, and Mona T Diab. 2013. Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media.. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics.*

[17] Dhruv Gupta, Jannik Strötgen, and Klaus Berberich. 2016. EventMiner: Mining Events from Annotated Documents. In *Proceedings of the 2016 ACM on International Conference on the Theory of Information Retrieval.*

[18] Prashant Gupta and Heng Ji. 2009. Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference.*

[19] Felix Hamborg, Soeren Lachnit, Moritz Schubotz, Thomas Hepp, and Bela Gipp. 2018. Giveme5W: Main Event Retrieval from News Articles by Extraction of the Five Journalistic W Questions. In *International Conference on Information.* Springer, 356–366.

[20] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Trans. on Knowl. and Data Eng.* (2009).

[21] Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics.*

[22] Heng Ji and Ralph Grishman. 2008. Refining Event Extraction through Cross-Document Inference. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics.*

[23] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. 2010. What is Twitter, a Social Network or a News Media?. In *Proceedings of the 19th international conference on World wide web (WWW).*

[24] Qi Li, Heng Ji, and Liang Huang. 2013. Joint Event Extraction via Structured Prediction with Global Features.. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics.*

[25] Q. Li, A. Nourbakhsh, S. Shah, and X. Liu. 2017. Real-Time Novel Event Detection from Social Media. In *Proceedings of the IEEE 33rd International Conference on Data Engineering (ICDE).*

[26] Q. Li, S. Shah, X. Liu, A. Nourbakhsh, and R. Fang. 2016. Tweet Topic Classification Using Distributed Language Representations. In *Proceedings of the 2016 IEEE/WIC/ACM International Conference on Web Intelligence.*

[27] Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.*

[28] Xiaomo Liu, Quanzhi Li, Armineh Nourbakhsh, Rui Fang, Merine Thomas, Kajsa Anderson, Russ Kociuba, Mark Vedder, Steven Pomerville, Ramdev Wudali, et al. 2016. Reuters Tracer: A Large Scale System of Detecting & Verifying Real-Time News Events from Twitter. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM).*

[29] Xiaomo Liu, Armineh Nourbakhsh, Sameena Shah Quanzhi Li, Robert Martin, and John Duprey. 2017. Reuters Tracer: Toward Automated News Production Using Large Scale Social Media Data. In *Proceedings of the 2017 IEEE International Conference of Big Data.*

[30] Juha Makkonen et al. 2009. Semantic classes in topic detection and tracking. *PhD thesis, University of Helsinki, Department of Computer Science* (2009).

[31] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *arXiv preprint arXiv:1301.3781.*

[32] Minh-Tien Nguyen, Duc-Vu Tran, Chien-Xuan Tran, and Minh-Le Nguyen. 2016. Learning to summarize web documents using social information. In *28th International Conference on Tools with Artificial Intelligence (ICTAI).*

[33] Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint Event Extraction via Recurrent Neural Networks.. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.*

[34] Thien Huu Nguyen and Ralph Grishman. 2015. Event Detection and Domain Adaptation with Convolutional Neural Networks.. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics.*

[35] Armineh Nourbakhsh, Quanzhi Li, Xiaomo Liu, and Sameena Shah. 2017. "Breaking" Disasters: Predicting and Characterizing the Global News Value of Natural and Man-made Disasters. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

[36] T. Nugent, F. Petroni, N. Raman, L. Carstens, and J. L. Leidner. 2017. A Comparison of Classification Models for Natural Disaster and Critical Event Detection from News.. In *Proceedings of the Workshop on Data Science for Emergency Management.*

[37] Miles Osborne and Mark Dredze. 2014. Facebook, Twitter and Google Plus for Breaking News: Is There a Winner?. In *Proceedings of The International AAAI Conference on Web and Social Media (ICWSM).*

[38] Ozer Ozdikis, Halit Oğuztüzün, and Pinar Karagoz. 2016. A survey on location estimation techniques for events detected in Twitter. *Knowledge and Information Systems* (2016).

[39] Jakub Piskorski, Hristo Tanev, Martin Atkinson, Eric Van Der Goot, and Vanni Zavarella. 2011. Online news event extraction for global crisis surveillance. In *Transactions on computational collective intelligence V.* Springer.

[40] Alan Ritter, Oren Etzioni, Sam Clark, et al. 2012. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining.*

[41] Polina Rozenshtein, Aris Anagnostopoulos, Aristides Gionis, and Nikolaj Tatti. 2014. Event detection in activity networks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining.*

[42] Parang Saraf and Naren Ramakrishnan. 2016. EMBERS autogsr: Automated coding of civil unrest events. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

[43] Hristo Tanev, Jakub Piskorski, and Martin Atkinson. 2008. Real-time news event extraction for global crisis monitoring. *Natural Language and Information Systems* (2008).

[44] Zhongyu Wei, Yang Liu, Chen Li, and Wei Gao. 2015. Using Tweets to Help Sentence Compression for News Highlights Generation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics.*

[45] Bishan Yang and Tom Mitchell. 2016. Joint Extraction of Events and Entities within a Document Context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics.*