Event Extraction from Italian Tweets: A Graph-Based Analysis

Luciano Imbimbo, Giovanni Bonisoli and Laura Po

[1]DIEF, Università degli Studi di Modena e Reggio Emilia

Abstract

Social media platforms, particularly X (ex Twitter), have become valuable sources of real-time information about real-world events. This thesis explores the application of graph-based methodologies for event extraction from Italian tweets, following the paper [1]. The study aims to:

1. ***Examine and test existing graph-based approaches for event extraction from tweets, evaluating their effectiveness in disambiguating tweets referring to the same event.***
2. ***Apply these methodologies to a corpus of Italian tweets to identify and categorize extractable events.***
3. ***Assess the accuracy and reliability of these methods in the context of the Italian language***
4. ***Explore potential extensions and improvements to existing approaches, considering the linguistic and cultural peculiarities of Italian tweets.***

This study aims to contribute to research on text mining and social media analysis in the Italian language, providing insights into the potential and challenges of extracting events from tweets for public safety and crime analysis purposes.

Introduction

In the age of social media, Twitter has emerged as a powerful platform for real-time information sharing, covering topics from every corner of the world. This constant stream of user-generated content has become an invaluable source of timely information about real-world events. However, the volume of tweets, combined with their unstructured nature and the presence of noise, poses significant challenges for extracting meaningful event-related information. Event detection from social media streams, particularly Twitter, has gained considerable attention in the Natural Language Processing (NLP) community. The task involves automatically identifying and clustering tweets that describe specific real-world events, ranging from major global events to localized incidents. Successful event detection can have wide-ranging applications, from news summarization and trend analysis to emergency response and public safety. Existing approaches to event detection on Twitter have primarily focused on two main strategies:

- Detecting spikes in clusters around specific keywords or Named Entities (NEs).
- Modeling relationships between terms contained in tweets using graph representations.

However, these methods face several limitations. Indeed, Keyword and NE-based approaches often struggle to distinguish between events that involve the same entities or similar terminology. Graph-based methods, while promising, can generate overly dense graphs and may mistakenly consider trending terms not related to events as event candidates. This thesis proposes a novel unsupervised approach to open-domain event detection from Twitter, addressing these limitations through a unique combination of Natural Language Processing techniques and graph theory. Our method represents the stream of tweets as event graphs, modeling the relations between Named Entities and the terms that surround their mentions in the tweets. By applying graph partitioning techniques and a modified PageRank algorithm, we aim to effectively cluster tweets related to the same event while distinguishing between similar events and filtering out non-event related content. The key contributions of this research include:

- A graph-based representation of tweet content that captures the contextual relationships between Named Entities and surrounding terms.
- An unsupervised algorithm for detecting and clustering fine-grained events without prior knowledge of the number or types of events.
- Extensive evaluation on a gold standard dataset, demonstrating good performance in event detection and clustering.

In the following chapters, we will detail the theoretical foundations of our approach, describe the methodology and implementation, present our experimental results, and discuss the implications and potential future directions of this research. Through this work, we aim to advance the field of event detection in social media and provide a robust tool for

extracting valuable insights from the ever-growing stream of social media data.

Approach Description

Preprocessing

The preprocessing stage of our approach is crucial for ensuring the quality and consistency of the input data. Unlike the method described in the paper, our preprocessing pipeline includes several additional steps. We begin by removing retweets,user mentions, URLs, and non-ASCII characters. We also implement a custom function to segment hashtags, breaking them into constituent words to preserve their semantic value. This segmentation is particularly valuable for capturing event-related terminology that might otherwise be lost in concatenated hashtags.
Our approach handles both English and Italian tweets, demonstrating its flexibility across languages. We employ language-specific date parsing to standardize timestamp formats, crucial for accurate temporal analysis of events. The cleaning process also includes the removal of emoticons and excessive whitespace, further normalizing the text.
For named entity recognition (NER), we diverge from the paper's use of NERD-ML and instead we use the most famous NLP toolkits, used in Python: Stanza, NLTK, and spaCy. Stanza's neural network-based models offer state-of-the-art performance in NER tasks across multiple languages. We complement this with NLTK's rule-based approaches and spaCy's efficient statistical models, creating a robust ensemble for entity recognition. This approach allows us to accurately identify and classify named entities in both English and Italian tweets.

Graph Creation

The graph processing stage builds upon the concept presented in the paper, where relationships between named entities and surrounding terms are modeled in a directed graph. So, the event graphs is built as follows:

- **Nodes:** We consider NE and k terms that precede and follow their mention in a tweet as nodes, where k > 1 is the number of terms surrounding a NE to consider while building the NE context.
- **Edges:** Nodes in the graph are connected by an edge if they co-occur in the context of a NE.
- **Weight**: The weight of the edges is the number of co-occurrences between terms in the NE context.

However, our implementation leverages a combination of Python and Neo4j, a graph database management system, to create and manipulate the event graphs. This hybrid approach allows us to harness the computational efficiency of Python for data processing and the powerful graph querying capabilities of Neo4j. By using Neo4j, we can efficiently store, retrieve, and analyze complex relationships between entities and terms, potentially uncovering more nuanced event structures than traditional relational databases would allow.

Graph Partitioning

After the definition of the graph we utilize graph theory to divide the graph into sub-graphs, which will be viewed as potential events. Tweets that pertain to the same events often contain several shared keywords [2]. This occurrence is represented in event graphs by stronger connections between nodes associated with the same event. In other words, the weights of edges linking terms from tweets about similar events are greater than those connecting terms from tweets regarding different events. The goal of the graph partitioning is to pinpoint such edges that, if eliminated, will separate the large graph G into sub-graphs.
To do that we use the built-in function of Neo4j. In particulary, we use the **Louvain**[3] hierarchical clustering method, an algorithm to detect communities in large networks. It maximizes a modularity score for each community, where the modularity quantifies the quality of an assignment of nodes to communities. This means evaluating how much more densely connected the nodes within a community are, compared to how connected they would be in a random network.

Event Detection

We assume that events from different sub-graphs are not related to each other. So, in the event detection sub-module, each sub-graph is processed separately. As proposed in the main paper and thanks to a study on local partitioning [4] we know that a good partition of a graph can be obtained by separating high-ranked vertices from low-ranked ones, if the nodes in the graph have distinguishable values. To rank and assign values to the vertices of the event-graph we use a PageRank-like algorithm:

$$S(V_i) = (1-d) + d \sum_{j:V_j \in IN(V_i)} \frac{\omega_{ij}}{\sum \omega_{jk}} S(V_j)\epsilon_i$$

where ω_{ij} is the weight of edge connecting V_i to V_j, d is a damping factor usually set to 0.85 [5] and ϵ_i a penalization parameter for node i. Following the paper, we define the penalization parameter of a node

according to its tf-idf score. Due to redundant information in tweets, the score of the nodes can be biased by the trending terms in different time windows. Thus, we use the tf-idf score to reduce the impact of trending terms in the collection of tweets. Before computing the score with equation 1, we assign an initial value $\tau = 1/n$ to each vertex in the graph, where n is the total number of nodes in the graph. Then, for each node, the computation iterates until the desired degree of convergence is reached. The degree of convergence of a node can be obtained by computing the difference between the score at the current iteration and at the previous iteration, which we set to 0.0001 [5].

As shown in the Algorithm 1 (image below), the initial step involves partitioning the vertex set into high-ranked and low-ranked vertices, guided by a defined parameter α. Subsequently, the process focuses on the high-ranked subset, starting with the highest-ranked vertices. For each candidate, the algorithm selects the highest-weighted predecessors and successors as keywords for potential event candidates (Lines 4-9). Following the removal of edges associated with these keywords from the graph, if the resultant graph becomes disconnected, the disconnected nodes are also considered as keywords for the event candidates (Lines 10-13).

Furthermore, utilizing the semantic classes identified by the Named Entity Recognition (NER) tool, the keywords pertinent to an event are categorized into specific subsets: "what" (denoting the type of event), "where" (indicating the location of the event), and "who" (referring to the individuals or organizations involved). In terms of the date, the oldest tweets reporting the event are selected.

In the subsequent phase of Algorithm 1, the refinement of event candidates is undertaken. Initially, the algorithm merges duplicate event candidates (Lines 22-35), specifically those that share common terms and possess identical locations or participants within the defined time window. A new event is then constructed from the union of terms and entities associated with the two event candidates. An event is considered valid if it involves at least one named entity and is reported in a minimum number of tweets, as stipulated by the input parameters.

Algorithm 1 Algorithm to process a given event-graph to retrieve important sub-events

1: **function** GRAPH_PROCESSING(G, α)
2: $E = \emptyset$
3: $H = \{v_i \in vertex(G) | score(v_i) >= \alpha\}$
4: **while** $H \neq \emptyset$ **do**
5: $G' = G$
6: $v_i = H.pop()$
7: $p = max\{W_j \in In(v_i)\}$
8: $s = max\{W_j \in Out(v_i)\}$
9: $keywords = set(p, v_i, s)$
10: $G'.remove\_edges(p, v_i, (v_i, s))$
11: **if** $\neg G'.connected()$ **then**
12: $append(keywords, disc\_vertices(G'))$
13: **end if**
14: $who = person | organization \in keywords$
15: $where = location \in keywords$
16: $what = keywords - who - where$
17: $tweets = tweet\_from(keywords)$
18: $when = oldest(tweets.date)$
19: $event = \langle what, who, where, when \rangle$
20: $append(E, event)$
21: **end while**
22: **for** $e \in E$ **do**
23: **for** $e' \in E$ **do**
24: **if** $what(e) \cap what(e')$ **then**
25: **if** $who(e) \cap who(e')$ **then**
26: $merge(e, e')$
27: **end if**
28: **end if**
29: **if** $where(e) \cap where(e')$ **then**
30: $merge(e, e')$
31: **end if**
32: **end for**
33: **if** $\neg who(e) \vee \neg where(e)$ **then**
34: $discard(E, e)$
35: **end if**
36: **end for**
37: **return** E
38: **end function**

Event Merging

At the end we perform a merge between the same events to avoid duplication. We consider events in different time-windows as duplicate if they contain the same keywords, entities in an interval of k days, where k is an input parameter.

Result

Dataset

We manually constructed two parallel datasets to enable cross-lingual comparison between Italian and English event detection capabilities. Each dataset consists of 80 carefully selected tweets, evenly distributed across four significant events (20 tweets per event).

- **Italian Dataset**, that covers the following events:
  - Salvini Open-Arms Trial
  - Monza-Inter Soccer Match
  - Sharon Verzeni Murder Case
  - Big Brother Italy
- **English Dataset**, that covers the following events
  - COVID-19 Global Pandemic
  - FIFA World Cup 2022
  - Iranian Revolution
  - COVID-19 Vaccination Campaign

To ensure data quality and consistent event association, we implemented a hashtag-based collection strategy. Each tweet in both datasets contains at least one event-specific hashtag, providing a reliable anchor point for event identification. While this approach slightly simplifies the typically unstructured nature of tweets, it enables more accurate keyword extraction and event association.
Then, both datasets were manually annotated to create parallel gold standards, maintaining identical structural characteristics across languages. This careful curation enables direct performance comparison between Italian and English event detection.

Experimental Setting

An event is considered correctly classified if all tweets within its cluster correspond to the same event in the gold standard. Conversely, if any tweets from different gold standard events appear in the cluster, the event is marked as misclassified. Given the relatively small dataset size, we established a gauged parameter $\alpha = 0.5$ as the minimum score threshold for nodes in the graph to be considered significant for event detection. To determine the tweet-cluster membership, following the methodology proposed in [6], we employ the cosine similarity. More in detail, we calculate cosine similarity between tweet words and community vocabulary, apply a threshold value (set to 0.2) to filter tweets and select only highly similar and therefore more significant tweets. Thanks to this process, we were able to define the accuracy metric as following:

$$accuracy = \frac{number\_of\_correctly\_classified\_events}{total\_number\_of\_events}$$

This metric enables us to quantitatively assess classification performance, compare results across different datasets and evaluate the effectiveness of our approach in both Italian and English contexts. The accuracy measure provides a straightforward way to evaluate the algorithm's performance in identifying and clustering event-related tweets while maintaining the semantic coherence of each event cluster.
Another metric used during the evaluation phase is the difference between the number of events in the gold standard (4) and the number of different event extracted by the clustering and event detection algorithm proposed. Thanks to that, we can understand how much the algorithm is able to disambiguate between same events.

NER Package Comparison

Before evaluating performance on Italian language, we investigated performance using different NER packages available in Python: Stanza, Spacy, and NLTK. This analysis aimed to determine which NER tool performs better on Italian language.

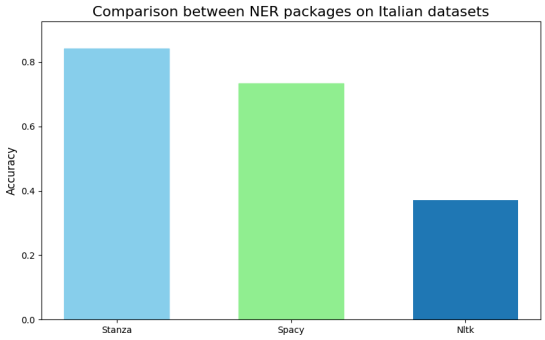

Figure 1: *Comparison between NER packages on Italian dataset*

The results demonstrate that Stanza achieves the highest accuracy for named entity recognition. This superiority is also evident in terms of disambiguation capabilities.
Indeed, as shown in Figure 1, Stanza (along with Spacy) minimizes the disparity between groundtruth and number of extracted events.
This evaluation highlights how the performance of this algorithm, and graph-based algorithms in general, significantly depends on the preprocessing performed on the initial dataset.
A common characteristic of these models is that performance varies significantly across languages.

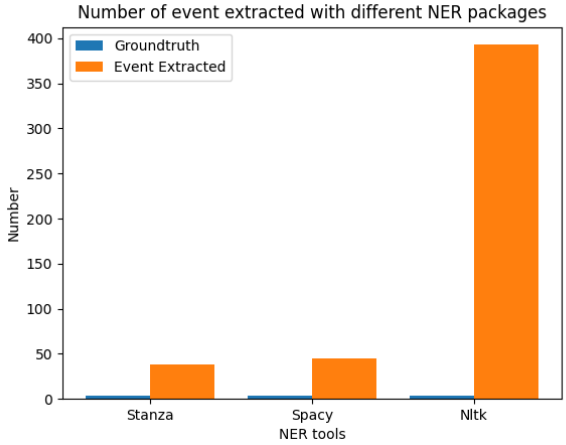

Figure 2: *Event extraction comparison across NER packages*

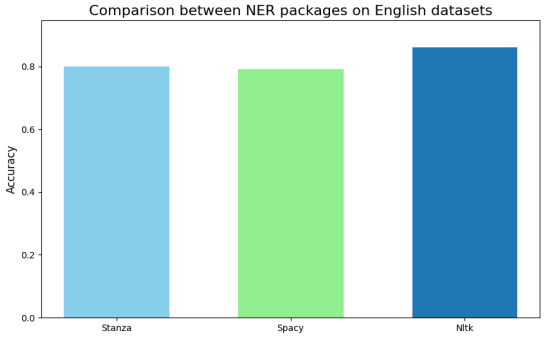

Figure 3: *Comparison between NER packages on English dataset*

In fact, the situation reverses for English, where NLTK achieves the best performance in terms of accuracy.

Performance on Italian Dataset

Based on the above findings, we opted to use Stanza's NER capabilities to evaluate performance on the Italian dataset. As shown in Figure 1, the algorithm achieves an accuracy exceeding 80% on the Italian dataset.
Since we cannot base our evaluation on a single metric, we proceeded with a qualitative analysis of the algorithm.
The first evaluation examines the difference between the number of events in the dataset and the number of events extracted by the algorithm. The significant disparity observed can be attributed to two main factors:

1. The **unstructured nature** of tweets: Many tweets associated with a specific event lack the main Named Entities of that event, preventing the algorithm from correctly associating the tweet with the event.

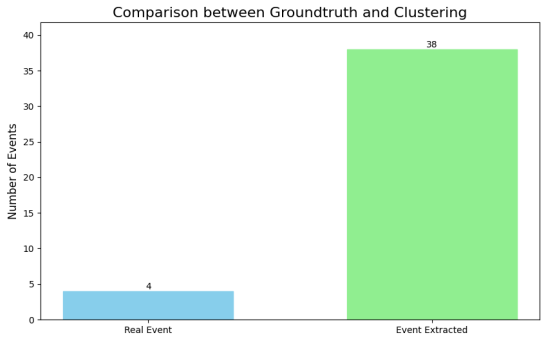

Figure 4: *Comparison between Groundtruth and Clustering*

2. **Entity disambiguation** problems: The algorithm, from event-graph creation to event detection, cannot disambiguate between NEs referring to the same entity (e.g., "Matteo Salvini" and "Salvini" are recognized as different NEs despite referring to the same person). This leads to more nodes and disconnected components in the graph, resulting in more clusters and extracted events.

Another evaluation focuses on named entity categories. We investigated whether certain fields tend to be empty:

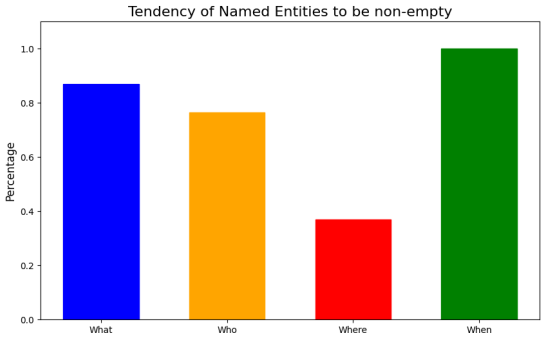

Figure 5: *Named Entities Empty Field Analysis*

The analysis reveals that only the 'where' category (event location) tends to be empty, primarily because most tweets, by their nature, lack specific location references.

Cross-Language Comparison

Finally, we conducted a comparative analysis of performance across different languages.
The differences are dramatic only with NLTK. However, when using models trained on multiple languages such as Spacy and Stanza, there are no substantial differences in algorithm performance across different languages in terms of accuracy.

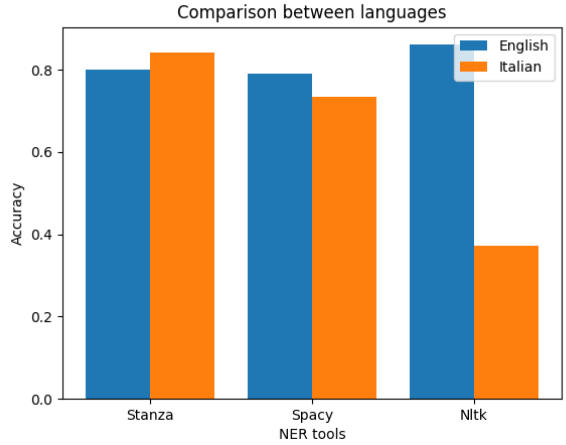

Figure 6: *Cross-Language Performance Comparison*

Conclusions and Future Works

In this paper, we described a model for detecting open-domain events from Italian tweets by modeling relationships between NE mentions and terms in a directed graph. The proposed approach doesn't reach adequate performance in term of accuracy and disambiguation. Possible improvements can help to reach better performance:

- Link terms to ontologies (e.g.DBpedia, YAGO) to help in detecting different mentions of the same entity and reduce the density of the event graph.
- Enrich the content of the tweets with information from external web pages resolving the URLs
- Use the hashtags and the most common NE associated to it to better detect events

References

[1] Amosse Edouard et al. "Graph-based Event Extraction from Twitter". In: (Sept. 2017). Ed. by Ruslan Mitkov and Galia Angelova, pp. 222–230. DOI: 10.26615/978-954-452-049-6_031. URL: https://doi.org/10.26615/978-954-452-049-6_031.

[2] Andrew James McMinn, Yashar Moshfeghi, and Joemon M. Jose. "Building a large-scale corpus for evaluating event detection on twitter". In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management* (2013). URL: https://api.semanticscholar.org/CorpusID:207206758.

[3] Neo4j. "Neo4j Documentation". In: (2024).

[4] Reid Andersen, Fan Chung Graham, and Kevin J. Lang. "Local Graph Partitioning using PageRank Vectors". In: *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)* (2006), pp. 475–486. URL: https://api.semanticscholar.org/CorpusID:206656317.

[5] Sergey Brin and Lawrence Page. "The anatomy of a large-scale hypertextual Web search engine". In: *Computer Networks and ISDN Systems* 30.1 (1998). Proceedings of the Seventh International World Wide Web Conference, pp. 107–117. ISSN: 0169-7552. DOI: https://doi.org/10.1016/S0169-7552(98)00110-X. URL: https://www.sciencedirect.com/science/article/pii/S016975529800110X.

[6] Hassan Sayyadi and Louiqa Raschid. "A Graph Analytical Approach for Topic Detection". In: *ACM Trans. Internet Techn.* 13 (2013), 4:1–4:23. URL: https://api.semanticscholar.org/CorpusID:11186546.