Network-based content geolocation on social media for emergency management

Supervisor: Prof. Barbara PERNICI
Co-supervisors: Prof. Chiara FRANCALANCI
Eng. Paolo RAVANELLI

Master thesis of:
Gabriele SCALIA  Matr. 838940

Academic Year 2015–2016
Abstract

This work is framed in the emergency management area, focusing in particular in the area of automatic social media information extraction to enhance the existing management processes.

The aim of the thesis is to present a new algorithm able to exploit the social networks naturally occurring during emergencies to automatically geolocate individual messages and show how this enables and enhances other tasks in the emergency management area.

The proposed algorithm is implemented and evaluated to demonstrate the usefulness of social networks to overcome the challenges in the automatic extraction of geographic information field given by the nature of individual messages which are often short, decontextualized and noisy.

Several case studies are used to demonstrate the usefulness of the extracted information in the context of typical applications related to emergency management processes: automatic event detection, situational awareness support and automatic image analysis.
Sommario

Questo lavoro si colloca nell’area della gestione delle emergenze, in particolare nell’estrazione automatica di informazioni dai social media per migliorare i processi di gestione esistenti.

L’obiettivo della tesi è quello di presentare un nuovo algoritmo in grado di usare le reti sociali che si creano naturalmente nei social media in concomitanza di eventi emergenziali per geolocalizzare automaticamente i singoli messaggi e mostrare come questo abiliti e migliori una serie di strumenti utili alla gestione delle emergenze.

L’algoritmo proposto è implementato e valutato al fine di dimostrare l’utilità delle reti sociali come strumento per superare i limiti nel campo dell’estrazione automatica delle informazioni geografiche imposto dai singoli messaggi presenti nel social media che sono spesso brevi, decontestualizzati e poco chiari.

Attraverso casi studio si dimostra poi l’utilità delle informazioni estratte nel contesto di diverse applicazioni tipiche nei processi di gestione delle emergenze: rilevamento automatico di eventi emergenti, supporto nel determinare lo stato dell’emergenza ed analisi automatica di immagini.
Acknowledgments

I would like to thank Prof. Barbara Pernici and Prof. Chiara Francalanci for their constant support and all the opportunities they gave and are giving to me and Eng. Paolo Ravanelli for his technical support and his advices.
# Contents

1 Introduction 1

2 State of the art 9
   2.1 Event detection in social media .................. 10
   2.2 Document-based event detection for new events on Twitter 12
   2.3 Tweets analysis .................................. 14
      2.3.1 Classification of tweets ...................... 16
         2.3.1.1 Domain adaptation ....................... 19
   2.4 Geolocation on Twitter ............................ 21
      2.4.1 Text-based methods ............................ 24
      2.4.2 Network-based methods ........................ 27
   2.5 Dimensions and open problems .................... 29

3 Geolocation 31
   3.1 Algorithm Overview ................................ 33
   3.2 Example ............................................ 38
   3.3 Local disambiguation .............................. 42
   3.4 Global graph management ......................... 46
   3.5 Global disambiguation ............................ 48
   3.6 Inference of missing locations ................... 51
   3.7 Discussion ......................................... 52
      3.7.1 Optimizations .................................. 52
      3.7.2 Limitations .................................... 55
      3.7.3 State diagram of a tweet ...................... 56
      3.7.4 Comparison with other techniques .............. 57

4 Processing pipelines and evaluation methods 61
   4.1 Phases overview ................................... 62
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1.1 Data acquisition and preprocessing</td>
<td>62</td>
</tr>
<tr>
<td>4.1.2 Data processing</td>
<td>65</td>
</tr>
<tr>
<td>4.1.3 Applications</td>
<td>66</td>
</tr>
<tr>
<td>4.2 Location-based pipelines</td>
<td>67</td>
</tr>
<tr>
<td>4.2.1 Location-based event detection</td>
<td>68</td>
</tr>
<tr>
<td>4.2.2 Rapid mapping and situational awareness</td>
<td>70</td>
</tr>
<tr>
<td>4.2.3 Image analysis</td>
<td>71</td>
</tr>
<tr>
<td>4.3 Evaluation methods</td>
<td>72</td>
</tr>
<tr>
<td>5 Experimental evaluation</td>
<td>75</td>
</tr>
<tr>
<td>5.1 Dataset</td>
<td>75</td>
</tr>
<tr>
<td>5.2 Parameter setting</td>
<td>76</td>
</tr>
<tr>
<td>5.3 Toponym recognition</td>
<td>76</td>
</tr>
<tr>
<td>5.3.1 Ground truth</td>
<td>77</td>
</tr>
<tr>
<td>5.3.2 Baselines</td>
<td>77</td>
</tr>
<tr>
<td>5.3.3 Proposed algorithm</td>
<td>79</td>
</tr>
<tr>
<td>5.3.4 Usefulness of the identified locations</td>
<td>79</td>
</tr>
<tr>
<td>5.4 Location disambiguation</td>
<td>81</td>
</tr>
<tr>
<td>5.4.1 Baseline</td>
<td>82</td>
</tr>
<tr>
<td>5.4.2 Proposed algorithm</td>
<td>83</td>
</tr>
<tr>
<td>5.4.3 Discussion</td>
<td>83</td>
</tr>
<tr>
<td>5.4.3.1 Geo/non-geo ambiguities</td>
<td>83</td>
</tr>
<tr>
<td>5.4.3.2 Geo/geo ambiguities</td>
<td>84</td>
</tr>
<tr>
<td>5.4.3.3 Algorithm limits</td>
<td>87</td>
</tr>
<tr>
<td>5.4.4 Statistics</td>
<td>88</td>
</tr>
<tr>
<td>5.5 Geolocation-based pipelines</td>
<td>89</td>
</tr>
<tr>
<td>5.5.1 Geotagged tweets vs text locations</td>
<td>89</td>
</tr>
<tr>
<td>5.5.2 Event detection pipeline</td>
<td>92</td>
</tr>
<tr>
<td>5.5.2.1 Classification</td>
<td>92</td>
</tr>
<tr>
<td>5.5.2.2 Temporal analysis</td>
<td>94</td>
</tr>
<tr>
<td>5.5.2.3 Location-based event detection</td>
<td>98</td>
</tr>
<tr>
<td>5.5.2.4 Discussion</td>
<td>104</td>
</tr>
<tr>
<td>5.5.3 Image analysis</td>
<td>105</td>
</tr>
<tr>
<td>5.5.3.1 Discussion</td>
<td>106</td>
</tr>
<tr>
<td>6 Conclusion and future work</td>
<td>109</td>
</tr>
<tr>
<td>Bibliography</td>
<td>119</td>
</tr>
</tbody>
</table>
Contents
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Example/1</td>
<td>39</td>
</tr>
<tr>
<td>3.2</td>
<td>Example/2</td>
<td>39</td>
</tr>
<tr>
<td>3.3</td>
<td>Example/3</td>
<td>40</td>
</tr>
<tr>
<td>3.4</td>
<td>Example/4</td>
<td>41</td>
</tr>
<tr>
<td>3.5</td>
<td>Example/5</td>
<td>41</td>
</tr>
<tr>
<td>3.6</td>
<td>State diagram of a tweet</td>
<td>57</td>
</tr>
<tr>
<td>4.1</td>
<td>Typical phases in a system for social media processing in the context of crisis management</td>
<td>63</td>
</tr>
<tr>
<td>4.2</td>
<td>Processing pipeline for location-based event detection</td>
<td>68</td>
</tr>
<tr>
<td>4.3</td>
<td>Processing pipeline for rapid mapping and situational awareness</td>
<td>70</td>
</tr>
<tr>
<td>4.4</td>
<td>Processing pipeline for image analysis</td>
<td>72</td>
</tr>
<tr>
<td>5.1</td>
<td>Image attached to a tweet: damages to a road</td>
<td>85</td>
</tr>
<tr>
<td>5.2</td>
<td>Local context built from the tweet</td>
<td>85</td>
</tr>
<tr>
<td>5.3</td>
<td>Global context built from the tweet’s neighbors</td>
<td>87</td>
</tr>
<tr>
<td>5.4</td>
<td>Cumulative number of tweets near Rome and near Accumoli according to the embedded geotags</td>
<td>90</td>
</tr>
<tr>
<td>5.5</td>
<td>Cumulative number of tweets near Rome and near Accumoli according to the extracted locations</td>
<td>91</td>
</tr>
<tr>
<td>5.6</td>
<td>Number of tweets per hour in the FLOOD dataset</td>
<td>92</td>
</tr>
<tr>
<td>5.7</td>
<td>Number of damage-related tweets per hour in the FLOOD dataset</td>
<td>94</td>
</tr>
<tr>
<td>5.8</td>
<td>Number of tweets per hour vs number of damage-related tweets per hour (scaled to have the same maximum)</td>
<td>95</td>
</tr>
<tr>
<td>5.9</td>
<td>The clusters formed the day 15/10/2016 with \textit{min}_\texttt{sample} = 10</td>
<td>101</td>
</tr>
<tr>
<td>5.10</td>
<td>The clusters formed the day 6/11/2016 with \textit{min}_\texttt{sample} = 10 and \textit{min}_\texttt{sample} = 20</td>
<td>102</td>
</tr>
</tbody>
</table>
5.11 The clusters formed the day 6/11/2016 with \textit{min\_sample} = 50 and \textit{min\_sample} = 75.
List of Tables

5.1 Precision, recall and F-score in the toponym recognition task . . . 78
5.2 Precision, recall and F-score in the toponym recognition task (useful
locations only). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 81
5.3 Comparison between F-scores: general case vs useful locations only 81
5.4 Accuracy in the disambiguation task . . . . . . . . . . . . . . . . . 82
5.5 Outcome of the classification . . . . . . . . . . . . . . . . . . . . . 94
5.6 For each day in the dataset: number of overall tweets, number of
damage-related tweets and percentage of damage-related tweets . . . 96
5.7 Outcome of DBSCAN location-based clustering for each of the three
days according to different values of min_sample . . . . . . . . . . . 100
5.8 The amount of geolocated images considering all the extracted im-
ages and only the useful and unique ones . . . . . . . . . . . . . . . . 106
List of Tables
Chapter 1

Introduction

This work is framed in the emergency management area, focusing in particular in the area of automatic social media information extraction to enhance the existing management processes.

Emergency management processes often handle crisis scenarios with the need to make quick decisions with minimal information about the event, like sudden natural disasters.

In this context, social media represent a unique and very helpful resource for situational awareness, that is understanding the “big picture” to gather insights during a natural hazard. In particular, microblogging social media like Twitter\(^1\) or Sina Weibo\(^2\) are the main resource adopted for this scope. Indeed, “microblogging is being considered as a means for emergency communications because of its growing ubiquity, communications rapidity, and cross-platform accessibility. This medium is also seen as a place for “harvesting” information during a crisis event to determine what is happening on the ground” \(^72\). Situational awareness means also coordinations, since it is “a collective intelligence process that involves many actors interacting with a combination of various source of information” \(^13\).

Therefore, information processing has a key role during crisis situations and social media communications represent a relatively new and increasingly important source to overcome information scarcity \(^53\,^72\). Indeed, during crisis people tend to use the systems more relevant for them, and social media are becoming more and more relevant.

Many messages are posted on social media right after an event and “studies show that these messages contain situational awareness and other useful informa-

\(^1\)http://www.twitter.com
\(^2\)http://weibo.com
1. Introduction

Information such as reports of urgent needs, missing or found people that, if processed timely, can be very effective for humanitarian organizations for their disaster response efforts” [62]. However, there are significant differences in information types among events because any disaster is unique and in some cases the most common information type of one crisis is absent in another, even if it is possible to individuate some trends in various types of crisis [53].

Processing social media messages is challenging: the millions of messages posted can be overwhelming and confusing and most social media posts do not include new and useful information but personal impressions or already known knowledge [53]. Information on social media is not verified and can be incomplete, since it is often de-contextualized in short and noisy messages. However, “some really interesting and important messages do get posted, sometimes providing information that is not available through other channels” [13].

Since most of the posted messages are not useful, manual analyzing them would require a disproportionate effort in terms of human work, and emergency organizations refer to this problem as “information overload”. However, the most appropriate term would be “filter failure”, since the problem is not having too much information, but not being able to filter only those messages which contain the actual useful information [13]. In this context, information technology has a fundamental role automating the processes and assisting human work in terms of efficiency and the effectiveness.

A key information is constituted by the locations mentioned in social media communications because attaching geographical coordinates to a message is useful for a number of other tasks in the context of crisis management [29] and an event is meaningful only if can be geographically characterized precisely [41].

On social media only a small minority of messages have machine-readable location information as metadata, that is geotags, due to a combination of constraints which include having a device capable to know the location (e.g. via Global Positioning System), an application capable to read it and the explicit user’s consent [13]. For example, on Twitter the percentage of geotagged messages is estimated between 0.5% and 2% [33, 40].

However, while metadata about locations may be often absent, many messages on social media contain references (implicit or explicit) to places [13]. Therefore, geolocation of messages through these references has an in immense value, not only in the emergency management context but also, for example, in text mining for business, marketing and defense applications [1, 33].

Different implicit and explicit location indicators exist, mainly: location men-
tions in the text, social networks, IP address, user-defined “location” fields, URL links and time zones [1].

Each of these indicators have different advantages and constraints and a certain applicability in terms of the type and the granularity of the target locations. Indeed, one can be interested to different types of locations on social media [1]: user location, that is the home residence of a user, posting location, that is the location where the message has been submitted, and content location, that is the location subject of the message content[3]. Depending on the application, a different location may be the main target: for example in the emergency management context the target could be the posting or the content location, since organizations are typically interested in messages posted near or about the affected areas, while in the business context a company could be interested in the user location for targeted advertising.

There is not a direct relationship between indicators and location types, in general. Most research has focused on locations mentioned in the text, because this field lays the foundation on traditional natural language processing (NLP) techniques. Social networks represent a peculiar characteristic of social media and have been used in particular to infer the user home residence. The user-inserted location field is useful to identify the user home residence or the posting location, depending if it is specified for the user’s profile or the single message, however it could be not directly machine-readable since it is inserted by the user itself; for example, on Twitter, it is a free text and therefore could be not a real location, could be ambiguous, misspelled, etc. Time-zones are a resource for coarse-grained geolocation and have the advantage of being always available when attached to the timestamps of messages. Of course, the geotags identify always the posting location.

The focus of this thesis is on locations mentioned in the text and social networks.

Extracting location references from text poses several challenges first of all in identifying the location names in the text (Named Entity Recognition, NER), since often they share the names with common words, have alternative names or are abbreviated and then in disambiguating them (Named Entity Linking, NEL) since many places in the world share the same name or part of it (e.g. London, UK and London, ON, Canada).

Locations mentioned in text are typically related to the message content or the

[3]Which could be distinguished, in turn, in the location directly mentioned in the message or described by its context.
posting location.

Techniques used traditionally in this field include supervised models, disambiguation clues from other locations mentioned in the text as well as clues from the gazetteer used to resolve those locations [78].

However, processing social media poses additional challenges since social media text is very different from traditional text like news articles because is less structured and de-contextualized, it employs abbreviations and often has typographical errors, repetitions, etc. Therefore, off-the-shelf systems for NER achieve poor results on social media [44, 60]. Indeed, for example, one of the most significant feature to recognize named entities, which is capitalization, is often absent in social media messages.

Network-based methods exploit the property that users on social media tend to interact mostly with the same people they interact in their lives, so this property is used to individuate the location of a user (or a tweet) based on the locations of his “friends”. Indeed, “in many cases, a person’s social network is sufficient to reveal their location” [61]. With these assumptions, most of the current research in network-based geolocation on social media is focused on user geolocation.

This thesis present a new algorithm able to exploit the social networks naturally occurring during emergencies to identify and disambiguate the locations mentioned in individual messages and then show how this enables and enhances other tasks in the emergency management area.

The proposed algorithm is both network-based and text-based. It focuses on identifying and disambiguating the locations in the text and, to do it, it uses the social network. Moreover, it tries to infer the location of those messages without explicit mentions in the text only using the social network and the other messages with mentioned locations.

Differently from most of the other existing techniques for network-based geolocation [16, 22, 36, 61], which address user geolocation and are based on explicit (articulated) social networks, like friendship, the proposed algorithm addresses message geolocation focusing on the implicit (behavioral) social networks. Indeed, social media are characterized by two kinds of social networks: explicit (articulated) social networks and implicit (behavioral) social networks [13]. Explicit social networks are based on codified relationships, like the “friendship” or the “following” relationships, while implicit social networks are inferred from communication patterns. As reported in [13], “implicit networks are particularly interesting in the crisis scenario because many exchanges happen among people who were not connected before the crisis.”
Therefore, the algorithm shifts the target of network-based geolocation from users to messages, and, to do it, shifts the focus from articulated to behavioral networks. The idea is not having a social network of friends and assume they live in near locations, but having a social network of messages behaviorally related (for example part of the same conversation or about the same topic at the same time) and assume they refer to related locations.

Focusing on behavioral networks and messages instead of users, new challenges arise. Mainly, behavioral networks are highly dynamic and new messages are continuously posted. Therefore, the algorithm must build an incremental graph of messages which stores their mutual relationships and use it to provide a context to each individual node, and all this must be done online as new messages arrive. The context allows to identify and disambiguate the locations mentioned in the messages and can be used also to infer a location in case a message does not have explicit references to names of places.

The locations related to messages are particularly useful and allow to overcome many limitations of user home locations in the emergency management context. First of all, there is not a direct relationship between the home residence of a user and the locations where he submits his posts or subject of his messages. Secondly, messages include many kinds of information and multimedia data (like images) potentially related to the current emergency situation that can be extracted, and message-level locations allow to directly localize them. Thirdly, messages can be collectively used in many emergency-related applications, for example for event detection searching for burst of posts related to certain topics or coming from specific locations, and message-level locations enhance these tasks.

The proposed algorithm is based on the fact that, even if a single message is decontextualized and noisy, its surrounding social network provides a context able to overcome these limitations.

The algorithm uses specific features of emergencies (the natural creation of behavioral networks with a huge number of messages posted in a short timeframe about the event) to overcome specific challenges of social media geolocation (lack of context, ambiguous and noisy messages) and address specific needs of emergency management (message-level locations). Therefore, the algorithm is not general purpose, but aims to exploit the characteristics of the domain to overcome its challenges.

The proposed algorithm is implemented and evaluated quantifying its performance in terms of precision, recall and accuracy processing the messages posted on Twitter during the earthquake that hit central Italy in August 2016. The re-
1. Introduction

Results demonstrate its effectiveness in identifying and disambiguating precisely the locations mentioned in messages.

Since geographical information has a key role for many application in the emergency management area, the proposed algorithm is then inserted in several processing pipelines to perform typical tasks required by crisis scenarios: event detection, maps supporting situational awareness and image analysis. Indeed, in the context of emergency management, geolocation is typically an enabler or an enhancer for other applications, addressing the limitations of geotags in terms of volume and location type and, if on one side geolocation can be useful by itself, for example to populate the maps of a live-monitoring system for social media, on the other side it is typically part of more articulated systems and its effectiveness in crisis scenarios is indirect.

These pipelines are evaluated through some case studies demonstrating the usefulness of text-extracted locations to overcome the limitations of geotags.

The motivation for this work originates from the the E²mC H2020 European project “which aims to demonstrate the technical and operational feasibility of the integration of social media analysis and crowdsourcing within the full Copernicus EMS (Emergency Management System), which provides annotated maps as a product, starting from satellite and aerial images” [20].

The thesis is structured as follows:

Chapter 2 describes the state of the art, focusing on the current techniques for event detection and text analysis on social media — in particular toponym recognition and location disambiguation — in the emergency management context. At the end of the chapter the main dimensions target of this thesis are outlined, focusing on the addressed open problems.

Chapter 3 presents the new algorithm for geolocation. An example precedes a formal description of the algorithm, which is then discussed in terms of optimizations, limitations and is compared with other existing techniques.

Chapter 4 proposes several processing pipelines corresponding to typical applications in the emergency management field where geolocation has a key role: event detection, maps supporting situational awareness and image analysis. For each one, the role of a geolocation phase — like the one performed by the proposed algorithm — is described. At the end of the chapter the evaluation methods and metrics are outlined.

Chapter 5 contains the results of the experimental evaluation. The proposed algo-
The algorithm is evaluated in terms of precision, recall and accuracy and is compared with other techniques, discussing the results. Moreover, the algorithm is evaluated in the context of the processing pipelines described in the previous chapter to show its effectiveness in typical applications in the emergency management area through some case studies.

Chapter 6 concludes the thesis and exposes possible future developments.

Part of this thesis has contributed to the paper [19], which has been accepted for publication at IEEE RCIS 2017.
1. Introduction
Chapter 2

State of the art

Social media text is very different from traditional text like news articles because is less structured and de-contextualized, it employs abbreviations and often has typographical errors, repetitions, and a unique language made by short messages with external links, images, emoticons, trending topic, irony and sarcasm and is characterized by a high level of noise. It is very difficult for traditional natural language processing (NLP) tools to handle well such text [60] and the information retrieval tasks are hampered by the fact that important information is regarded as noise.

For these reasons, even if the techniques employed to analyze social media are based on those for traditional media, ad-hoc methods and solutions have been proposed to handle the specificities.

Moreover, crisis scenarios, like natural disasters, pose further constraints and challenges with respect to other kinds of events [29].

The following state of the art is focused on social media analysis, in particular to support crisis management, with emphasis on natural disasters. Most of the presented papers are focused on Twitter, because it is the most analyzed social media in this context[1] but the discussion will be general.

This chapter starts with an overview of event detection in social media in Section 2.1 with a focus on document-based event detection in Section 2.2. The main kinds of analysis typically performed on social media, in particular supervised classification, are in Section 2.3. The state of the art about geolocation on social media, with emphasis on disaster management and crisis scenarios, is detailed in Section 2.4 since the new algorithm presented in Chapter 3 focuses on geolocation.

[1] Differently from Twitter, most large social media platforms does not offer its level of data access publicly [29].
in this context. At the end of the chapter, in Section 2.5 the main dimensions subject of this thesis and the open problems are outlined.

## 2.1 Event detection in social media

There are many definitions of events depending on the context and the use case. In most cases events happen in a precise time and in a delimited space \[3,70,74,76\]. Events can be real-world events or social events, can be periodic, announced or unexpected and can vary a lot in terms of time and space.

The *Topic Detection and Tracking* (TDT) research program \[3,76\] traditionally has addressed the event detection problem on the conventional media sources.

Although event detection on social media lays the foundation on the traditional event detection, there are new challenges in different fields, like:

- the presence of more informal, irregular and less structured content;
- scalability, since social media are an example of big data;
- data acquisition, since often the available data provided by the platforms is limited or inaccessible directly;
- the characteristics of the media, which continuously produce new content, and the events themselves, which may need rapid responses (for example in the case of mass emergencies), can demand online and incremental algorithms.

Mass emergencies are specific cases of real-world, unexpected events, and they pose new additional challenges \[37\]; natural hazards are specific cases of mass emergencies. In particular, crisis situations call for timely, reliable information to act in a very uncertain environment.

Event detection on social media is an interdisciplinary field which comprises techniques from many research areas, such as machine learning, natural language processing (NLP), data mining and text mining.

Techniques for event detection, in general, can be divided into document-pivot and feature pivot techniques \[8\]:

- Document-pivot techniques: the focus is on the documents (e.g.: news articles, tweets), to discover new events, tracking them and check when they expire based on grouping and segmenting the documents themselves. Each
document is represented in a space as a vector, using a transformation like the classical term frequency-inverse document frequency (tf-idf) approach or other schemes that for example privilege entity words, take into account time, space or the underlined topic of the document. Then, the documents are grouped with techniques like clustering and community detection on a graph. As a first approximation, an event can be seen as a cluster of documents.

- Feature-pivot techniques: the focus is on the features themselves (e.g. the number of certain keywords) to observe how they vary on time and space, identify bursty areas and new trends. Therefore an event can be seen as a set of features (like keywords) showing burst in appearance counts.

Therefore, in the first case the focus is on the documents themselves and it is useful when it is necessary to extract information (e.g. images, metadata) from them singularly; in the second case the focus is on the features which characterize the event as a whole.

Different kinds of event detection exist: retrospective event detection (RED) versus new event detection (NED), unspecified versus specified event. Each choice brings its constraints and challenges: an overview of the different kinds of event detection techniques is detailed in the surveys [8] and [29].

A system for crisis management performs NED since the aim is to discover new events as soon as they happen rather than analyzing events occurred in the past\(^2\) and focuses on specified events (e.g. natural disasters). Therefore, in the following the focus will be on this kind of event detection.

As reported in [29], “the most useful NED systems for emergencies are those who perform this analysis on-line”, that is processing a new document as soon as it arrives.

Moreover, document-based techniques will be privileged because they allow to process the documents as meaningful instances to extract information, media and locations from each document separately, besides identifying and tracking the event itself\(^3\).

\(^2\)Many systems described in the literature perform NED even if they are evaluated on datasets acquired in the past: this can be done simulating the arrival of messages using the original timestamps.

\(^3\)Indeed, the output of a document-based event detection algorithm is not constituted only by the events themselves, but also by the set of documents part of each event.
2.2 Document-based event detection for new events on Twitter

In Twitter, each document is a tweet.

A typical way to perform document-based event detection is aggregating documents using (optionally online) clustering techniques. [10] uses a single pass incremental clustering algorithm to group similar tweets. There is a threshold parameter to decide if a tweet is part of an existing cluster or not. At each point in time, there is a set $A$ of active clusters. A tweet which cannot be included in any active cluster is considered itself a new cluster and a cluster in $A$ which becomes inactive is removed. Each tweet is represented as a $tf-idf$ vector (see Subsection 2.3.1) and the cosine similarity [39] is used as distance function among tweets. For performance reasons a centroid representation of the clusters is employed, so that each cluster is represented only by its centroid which is a summary of its tweets.

The paper [17] extends [10] including also semantic information to drive the clustering algorithm. This is obtained assigning a topic to each tweet using a Twitter-tailored variant of the Latent Dirichlet Allocation (LDA) algorithm, TwitterLDA [79], which assumes that a tweet, being short, has only one topic. Assigning a topic to each tweet, they observed a worse performance with respect to [10]; however, assigning a topic to each hashtag in an online fashion they were able to improve the performance with respect to [10]. This is because topics assigned to tweets are more fine-grained but also more noisy.

A limit of approaches based on incremental clustering which compare a new document to all the others is that the complexity increases as more documents have been processed and therefore such algorithms are not scalable. Comparing a new tweet to clusters represented as centroids like in [10] mitigate the issue but the resulting comparisons are not among documents themselves. [56] focuses on “first story detection” on Twitter and a different approach is employed: they use an algorithm based on locality-sensitive hashing (LSH) which assigns similar hashes to similar tweets and therefore is able to process any new tweet in constant time and constant space, keeping comparisons which are based on (the hashed version) of tweets and not on centroids. Other works follow and extend this strategy, e.g. [68].

A commonly used approach for event clustering is density-based clustering, like DBSCAN [18] and its extensions. In density-based clustering, a cluster is defined as an area with a high density of data points. The main advantages of DBSCAN are:
2.2. Document-based event detection for new events on Twitter

- can find arbitrarily shaped clusters;
- does not require the number of clusters as parameter, like for example K-means clustering;
- naturally handles outliers.

These characteristics make density-based clustering a valid approach for event identification on Twitter since many events do not have a predefined “shape” (both from the geographical and semantical point of view), the number of events (therefore clusters) is not known performing new event detection (NED) and on Twitter there is a high level of “noise” \[56\], that is not relevant tweets, which are outliers from an event-detection point of view.

\[9\] focuses on clustering tweets to discover specific events or aspects characterizing events according to user perception. Each tweet, after a preprocessing step, is represented as a tf-idf vector (see Subsection 2.3.1). The DBSCAN algorithm is then used for clustering. In particular, a multiple-level clustering approach is employed to take into account a variable density distribution of tweets in the events.

All the works cited up to now cluster tweets according to their message content (that is, the text and text-derived features), eventually representing the resulting events on a map. The consequence is that an event is not necessarily precisely identified geographically, therefore such approaches privilege a definition of event which happens in a certain time and have a similar text content but potentially could be something like a global event or could be a not-real-world event. However, in the context of crisis management, in particular for natural disasters, the geographical aspect is crucial and often an event is meaningful only if can be geographically characterized precisely \[41\], therefore the spatial aspect of the event becomes a requisite for its detection.

\[59\] describes several improvements over \[10\] and takes into consideration the geographical aspect of events. Instead of comparing a new tweet against all possible clusters, a candidate retrieval step is added to find the most promising clusters efficiently. The feature representation of each tweet is enriched with temporal and geographical information (using the timestamp and the coordinates associated to geotagged tweets) and to consider them another type of similarity is added near to cosine similarity.

\[41\] focuses on event detection, in particular for those events which can be precisely identified spatio-temporally, like natural disasters. In their framework a
first step cluster tweets in a sliding window from a content and temporal point of view, using an incremental version of DBSCAN, and a second step analyzes the tweets to estimate the locations of the events based on the concept of spatial locality, that is “a set of messages concerning some topic, which are highly densely located in a specific geographical area”.

69 proposes to extract bursty areas associated with local topics and events from geotagged tweets using a spatiotemporal clustering algorithm. Their algorithm is an extension of DBSCAN but differently from 9 or 41 the algorithm does not focus on textual features but on the locations themselves and on the time dimension. This means that the tweets are interpreted spatially to be clustered according to their actual geographical distance. 64 is an extension of 69 which also takes into account the semantic of tweets to privilege clusters with tweets sharing keywords.

2.3 Tweets analysis

In this section the main techniques for tweets analysis are outlined. Indeed, the processing pipelines for emergency management proposed in Chapter 4 will employ several kinds of analysis. Moreover, geolocation, which is the main topic of this thesis, is a special case of analysis, and for its importance the related state of the art will be detailed in the next section.

When a target set of tweets has been identified (for example all the tweets related to a certain event), it is possible to analyze them using their textual content or their metadata. This can be done in various ways, typical ones are information extraction, automatic summarization, semantic enrichment or supervised classification. Also geolocation is a type of tweet analysis, even if for its importance in this work it has been described in its own Section 2.4.

Information extraction is the task of extracting structured information starting from unstructured and noisy tweets. For example, in 71 this task has been addressed using linguistic patterns and supervised learning; in 30 it has been addressed applying conditional random field (CRF), a statistical model which predicts the class of a text token based on its context in the sentence. The information extraction task often needs to be tailored to the specific target information, hence it is generally suitable for domain-specific event detection systems.

Automatic summarization is the task of generating a summary from a set of tweets, which brings the maximum event coverage in the minimum text space, to deal with the abundance of messages on social media. Twitter is a very dynamic
environment and an event itself evolves during time, hence automatic summarization must be done in an incremental and temporal manner, and this is challenging. Both extractive (extracting a set of tweets which maximize the coverage) and abstractive (generating new sentences which capture the core information) have been used. For example [51, 67] use clustering followed by a ranking algorithm to perform extractive summarization identifying those tweets which better describe their own clusters. [52] use a graph built on bigrams (two consecutive words) which is updated incrementally in real-time and is capable of generating abstractive summaries selecting those words which better describe the event interdependently even if they do not occur together in the same tweet. Recently it has been shown that a two-step extractive-abstractive summarization strategy can improve results in terms of information coverage, diversity, redundancy, coherence, and readability [62]. In this context the main challenges are represented by scaling issues of the algorithms and by the fact that the relative importance of different features is not well understood, hence their development is still preliminary [29].

Semantic enrichment task is a broad field in social media analysis. It is often preceded by an information extraction phase. The aim is to deal with the variety of expressions which refer to the same semantic concept, linking together different expressions (with their related concepts). Traditionally, named entity linking (NEL) is a semantic task where the aim is to link those n-grams which refer to an entity (found through the named entity recognition (NER) task) to the exact and unambiguous entities they refer to in a knowledge base. Indeed, the same entity can be described by many surface forms and the same surface form can refer to different entities. In general, entities can be of various kinds (people, locations, companies, etc.), according to the context. In this respect, location disambiguation is a special case of NEL, as detailed in Section 2.4. Other kinds of semantic enrichment tasks are Twitter-specific, for example “hashtags are found to be an important semantic features for tweets, because they help identify the topic of a tweet and estimate the topical cohesiveness of a set of tweets” [8] and indeed hashtags are often used as semantic features to cluster or to classify tweets. For example, [73] discover word semantic relations even if words do not co-occur within a specific tweet, using a hashtag-based topic model. Topic modeling, for example using latent Dirichlet allocation (LDA), is another type of semantic enrichment. In this case a topic (or multiple topics) is assigned to the tweet (or to part of it) [48, 80].

One of the most used analysis task, especially for domain-specific events (that is, when the target event type is known), is supervised classification. In this case the idea is to assign to each tweet a class (that is, a category) using a classifier
which is previously trained on another set of tweets and their manually-assigned\textsuperscript{4} labels. Text classification is a broad field of research, both in itself and in the social-media context. For this reason and because it is employed in the pipeline for event detection described in Chapter 4 along to geolocation, the related state-of-the-art is described in the following subsection.

### 2.3.1 Classification of tweets

Supervised text classification is a broad field which lays the foundation on machine learning. In the traditional formulation, the aim is to assign each tweet to a class among a set of predefined ones, using a supervised classification algorithm, that is a classifier, trained on a training set, which consists of class-labeled tweets. The classifier learns a model which then can be used on the testing set, that is a set of unlabeled tweets.

In the context of event detection on Twitter, and in particular in the disaster-related domain, a classifier is typically employed to identify a subset of tweets, for example those related to a certain kind of event, written by a certain kind of user or which bring a certain kind of information. While some crisis-related ontologies exist, there is not a single widely accepted way of categorizing crisis-related social media messages \cite{29}; for example tweets have been classified according to their factual, subjective or emotional content, by the kind of information provided, by the kind of information source, by their credibility, by time and by location. The choice on the set of classes is driven by the needs of the system, but also by the target event itself, since there exists a “substantial variability across crises” \cite{53} that must be taken into account.

Different supervised classification algorithms have been applied in the context of crisis-related tweets, like naïve Bayes \cite{65,77}, support vector machines (SVMs) \cite{15,77}, logistic regression \cite{7}, decision trees and random forests \cite{32}. Moreover, recently also convolutional neural networks have been applied to twitter text classification in the crisis-related field \cite{12,50}.

A system which performs text classification typically goes through the following steps, which, in the social media context, are characterized by specific challenges:

**Preprocessing** This involves the tasks of normalizing the text before performing the feature extraction. Typically the stop-words (that is, the most common

\footnotetext[4]{More precisely, the labels could not be totally manually assigned, for example in the case of semi-supervised classification.}
words in a language) are removed since they are not significant in characterizing a message and all the other words can be normalized through a stemming/lemmatization algorithm (this means substituting inflectional and derivationally related forms of words with their “base form”, e.g.: am, are, is ⇒ be), since this can reduce the vector space with a positive impact on the classification task. Then, there are Twitter-specific tokens which must be handled since they carry a specific semantic meaning, like “RT” for a retweeted message, urls, mention (in the form of @username) and hashtags (in the form of #hashtag).

Features extraction This step is crucial in all classification tasks. Given a tweet in its textual form, it must be transformed into a vector in a certain vector space. This involves a feature selection phase in which the features are designed. Often the features traditionally used for text classification are employed; they include: unigrams (single words), bigrams (couples of consecutive words) and in general $n$-grams, part-of-speech tags\(^5\) and morphosyntactic features, besides statistical features like message length and punctuation occurrences. $n$-grams are typically represented using the traditional term frequency, where the vector space consists in all the words in all the tweets and for each document $d$ and each term $t$ the term frequency $tf(t, d)$ expresses how often $t$ occurs in $d$, or by the term frequency–inverse document frequency, where the term frequency is weighted by the inverse document frequency, that is a measure of how much a term is rare across documents and it is useful to weight less the most common words privileging the rare ones\(^6\).

In certain cases features are also added to better generalize the text, like the class of synonyms for a certain word, or features from other fields like the “sentiment” as defined in social media sentiment analysis. Feature selection is a key to domain adaptation (see \(2.3.1.1\)).

Classification The classification step itself assigns a class to each tweet. As said, different types of classifier and different set of classes have been employed in literature. In the following, several examples are reported.

Many papers describe a classifier used in this context, both as a standalone tool or part of a more complex system.

---

\(^5\)Part-of-speech tagging on Twitter is an open problem which poses additional challenges with respect to traditional text \(23\). Its discussion is out of scope.

\(^6\)Different variants of term frequency, inverse document frequency and tf-idf exist, like logarithmically scaled, normalized, to handle limit cases and using different weighting schemes.
An additional challenge, which is typical of any supervised classification method, is the availability of human-annotated data, since it requires time and money\cite{31}. However, with a domain adaptation approach, it is possible to use human-annotated data already available in the literature (e.g. \cite{15,31}) limiting the necessity of further human annotations.

The paper \cite{65} focuses on event detection on Twitter, specifically for earthquakes. They use a support vector machine (SVM) with two classes to detect tweets really related to earthquake and avoiding messages which only incidentally contain earthquake-related words. As features, they use the words in the message, the surrounding words for each word and statistical features about the message.

In \cite{77}, which uses natural language processing and data mining techniques to extract situation awareness information from Twitter messages generated during various disasters and crises, a binary classifier is trained to detect tweets about the disaster’s impact on infrastructure. As a preprocessing step, they remove stop words and apply tokenization. As features, they combine lexical features (unigrams, bigrams, etc.) with Twitter-specific features (number of hashtags, number of mentions, etc.). Both a naïve Bayes and a SVM classifier are trained.

\cite{30} focuses on automatic methods for extracting information from microblog posts. They first classify tweets based on their contribution to situational awareness distinguishing personal, direct and indirect informative messages; then, for the informative messages, they further classify them in different “information types” like “caution and advice”, “casualties and damage”, etc. As preprocessing step they removed stop words and non-words and performed stemming. As features they use lexical features, Twitter-specific features and also part-of-speech tags and Verbnet classes for each verb in the tweet (“Verbnet is an ontology for verbs that organizes a set of similar verbs into classes and consists of a hierarchy of verb classes that include hypernyms, synonyms, etc.”). A naïve Bayes classifier is used.

\cite{7} focuses on extracting actionable information from Twitter during natural disasters. As part of the system, they use a classifier to identify tweets reporting damage or casualties. Each tweet is represented as a bag-of-words and different classification algorithms are compared.

A recent challenge addressed in the literature in this context is domain adaptation. It must necessarily be addressed by real systems which have to handle different types of natural disasters and must be scalable to different languages, and it will be also a goal of the proposed pipelines (see Chapter\cite{4}). Therefore, the state of the art about domain adaptation in the context of classification of tweets during natural disasters is detailed in the following.
2.3. Tweets analysis

2.3.1.1 Domain adaptation

The domain adaptation problem is a traditional problem in the machine learning field, when it is necessary to train a model on a source data distribution to apply then it on a different (but related) data distribution.

In the context of learning models for exceptional events, in particular natural disasters, this problem is made worse by:

- The fact that often it is necessary to extract real-time insights, especially just after the event, to help the relief effort. This means that in many cases it is not possible, at least in the first hours, to rely on annotated data from the target event itself and only data about past events could be available\cite{15,32,42}.

- The fact that each natural disaster, being an exceptional event, it is an *unicum* and can be uniquely characterized in terms of time, space, languages involved, damages, etc. A natural disaster is a complex and rare event that often is handled with ad-hoc procedures, therefore from a machine-learning point of view the domain adaptation gap can be very significant\cite{32,42}.

These problems, in particular when a language domain-adaptation is involved, have been addressed only recently in the literature.

The paper\cite{15} focuses on training a classifier to detect those messages which carry critical information for the damage assessment task, distinguishing tweets in: those related and carrying damage information, those related but without any damage information and those not relevant. In particular, they study the domain adaptation problem (on the single Italian language) when a training set from a different type of disaster is used (e.g. training on a flood and testing on an earthquake) or when a training set from a different disaster but of the same type is used, investigating the features which achieve the best results. They annotated four datasets, two about floods and two about earthquakes, and then they used lexical features, morpho-syntactic features, lexical expansions and also sentiment analysis features to train a support vector machines (SVM) classifier. The results show that when both the training set and the test set come from the same disaster type it is possible to obtain good results using only low-level linguistic features, but when the training set is about a different disaster type additional syntactic and sentiment features are useful. Regarding the use of training sets obtained combining two events of different types, they reported mixed results, with a positive impact on some test sets and a negative impact on others.
In [32] a similar analysis is performed, but also the cross-language domain adaptation case is taken into account. In particular, the paper focuses on studying the usefulness of labeled data of past events to classify tweets belonging to new events. They selected 11 crisis consisting in 5 earthquakes and 6 floods, with the tweets annotated on the basis of the “information type” they carry. As preprocessing step they removed stop words and non-words, and they used only unigrams and bigrams as features to train a random forest classifier. Then, they performed different types of experiments: training on a single source event and testing on a target event, both in-domain (same kind of event) and out-domain (different kind of event) and training on multiple source events and testing on a target event, both with and without tweets belonging to the target event. The results highlight that considering the same type of event, “including more training data, even from a mixed language source, improves the accuracy significantly”, and that a combination of training data in languages different from the target event can outperform a training set in the same language of the target event if the latter has not enough data. However, they also show a real benefit in using training data from a different language only when the “lexical similarity” between the languages is high, like in the Italian-Spanish case, and not when the languages are significantly different like in the Italian-English case. Considering events of different types there are mixed and not conclusive results. Obviously, training the classifier using data from the target event always increases the performance.

[42] follows a different approach to domain-adaptation: they combine labeled data from past events with unlabeled data from the target event. As preprocessing step, they cleaned the tweets from not useful words and non-printable characters. They perform a classification based on three questions: Q1) if a tweet is about the specific disaster Q2) if it offers support for the victims Q3) if it expresses any emotion to the victims. They considered two events. As baseline, they trained supervised Naïve Bayes classifiers using only labeled data from the past event, and then they used a domain adaptation algorithm which exploits both labeled data from the past event and unlabeled data from the target event. The results show that for the question Q1 there is an improvement using unlabeled data, but not for Q2 and Q3. The motivation is that Q1 is a more event-specific question, while Q2 and Q3 are more general across events. Therefore, “experiments suggest that source data from a prior disaster can be used to learn classifiers for a current target disaster, especially for tasks that are similar across disasters. Furthermore, using source labeled data together with target unlabeled data in a domain adaptation framework has the potential to produce better classifiers for tasks that are more
specific to a disaster”.

## 2.4 Geolocation on Twitter

Only a small percentage of messages on social media is natively geotagged (e.g. about 0.5%-2% on Twitter [33,40]), however many messages posted on social media contain references to name of places [13]. Geolocation means assigning a location to something (for example a user or a content like a message). This can happen using machine-readable locations (trivial), inferring an unknown location starting from other indicators or recognizing a not machine-readable location (for example in a message or on the profile) disambiguating it.

Research into geolocation on Twitter ranges around different aspects in terms of type of location, granularity and precision of the identified result and also the focus varies from the single person, to a group of connected people up to everyone related to a certain event, since different types of information are required by different scenarios [1].

Different types of locations can be extracted from twitters: user home residence, that is the location of the home’s author of the message(s) [22,26,36,43,58,61,63], user location (or posting location), that is the location of the user when the tweet is posted [14], and the location(s) related to the message content [21,33,34,38,44,55,78]. The latter could be further divided in locations directly mentioned in the message text and locations described by the message context. Often a combination of these location types has been used in practical applications to maximize coverage, like in [40]. It is important to notice that the different location types are in general independent from each other and they are all independent with respect to the features used to find them: a location mentioned within a message does not imply that the user location is related [27] nor that the message content is related. However, in the applications often no distinction is made between different types or there is the assumption that a certain feature is always related to a certain type of location.

Different features have been used to identify locations based on twitters [1]: location mentions in twitter text, friend’s network, the location field of the tweet, website IP addresses, geotags, URL links and time zones.

A tweet comes bundled with a set of machine-readable metadata associated to the text content. Among them, different fields are location-specific:

- **location**: the location defined by the user for his account’s profile;
2. State of the art

- **geo_enabled**: when true, indicates that the user has enabled the possibility of geotagging their Tweets;
- **coordinates**: is expressed as a longitude-latitude pair and represents the location of the tweet as reported by the user or client application;
- **place**: it is a user-inserted field for the tweet which should indicate a place related to the content or to the actual position of the user.

Note that all these fields are nullable, location could be unparsable and could not be a real location and both location and place do not necessarily have an actual relationship with the message content or the actual position of the user. Considering global tweets randomly sampled, [40] reports that 41% of users have agreed to share their location at least once, 35% of the users have specified a location, only 2.5% of the tweets come with a not-null place field (which is at city-level granularity in 89% of the cases) and only 2% of the tweets are geotagged with precise location coordinates.

Since the processing pipelines in the emergency management context (Chapter 4) are typically interested in the locations related to the message content and the posting locations (as will be seen it is often hard to distinguish between them and they are both useful in the target domain), the focus in the following will be on the papers related to these types of locations. However, the novel algorithm for geolocation developed in this thesis and presented in Chapter 3 is also inspired by papers which use social and contextual links between users and tweets to geolocate them, and they often aim to find the user home residence; so they will be described even if the user home residence is not the focus of this work.

An important difference among researches in geolocation is between location disambiguation [11, 33, 55, 57, 78] and location inference [16, 36, 46, 47, 75]:

**Location disambiguation task** It aims to geocode an expression, i.e. finding the coordinates given a more or less qualified name. It is also called toponym resolution when it involves a disambiguation step [78]. The disambiguation is necessary because an expression could not be fully qualified, could not really be a location or could refer to many different locations in the world. This task is similar (actually, is a special case) of the named entity linking (NEL) task, which is a well-known and addressed topic in literature both in traditional text and social media [66]. Traditionally named entity linking follows named entity recognition (NER) [35], called also mention detection, which is the task of finding named entities
of various types (people, locations, companies, etc.) in a text. In the context of location entities, it is also called geoparsing. In the NEL task there is an knowledge base (or external source) of information (like DBpedia, Yago, Freebase), which stores the entities, their semantic classes and their mutual relationships. The choice on the external source depends on the context; for location entities GeoNames\textsuperscript{7} [21, 33, 44, 55, 78] and OpenStreetMap\textsuperscript{8} [22] are frequently used knowledge bases.

As reported in [66], “the entity linking task is challenging due to name variations and entity ambiguity. A named entity may have multiple surface forms, such as its full name, partial names, aliases, abbreviations, and alternate spellings” and “a named entity may have multiple names and a name could denote several different entities”. This highlights that entity disambiguation (or location disambiguation, in particular) and entity linking (or location linking, in particular) are actually the same thing called differently [78]. Moreover, even if traditionally NER and NEL tasks are considered two different stages of a pipeline, they can be merged in an end-to-end algorithm with improvements on both the stages [24, 34]. Indeed, being a pipeline, errors in the NER (or geoparsing) phase negatively affect also the NEL (or geocoding) phase. Traditionally, near to techniques commonly used in NLP like conditional random fields (CRF), also other locations mentioned in the text or related texts as well as clues from the gazetteer have been used to disambiguate a location.

**Location inference task** It aims to *discover* the location, based on features like the content, the relationships with other documents, the metadata, etc. In this case there could be no location mentions in the text, therefore the location must be inferred. Typically the connections of a document with other documents are exploited [16, 36, 47], however it is possible also to use the text content in a probabilistic way rather than searching for explicit location mentions [27]. Other solutions use language models or time zones [1].

In some cases location disambiguation and inference are addressed as a unique problem, like [22]. Also the algorithm presented in Chapter 3 addresses both disambiguation and inference (privileging the disambiguation task in the current implementation).

Many algorithms for geolocation have been proposed. They can be categorized from different point of views, like the type of location they disambiguate/infer or the type of feature they use [1]. In the following they are classified in *text-based*...

\textsuperscript{7}http://www.geonames.org
\textsuperscript{8}http://www.openstreetmap.org
2. State of the art

methods, that is algorithms which use the tweet content itself, and network-based methods, that is algorithms which exploit social networks. Also hybrid approaches exist \cite{22,58} and the algorithm presented in Chapter \ref{chap:methods} is indeed both text-based and network-based\footnote{The algorithm is network-based, but the goal is to disambiguate text-extracted locations.}

2.4.1 Text-based methods

In this case the challenges are: the unstructured, error-prone and noisy nature of tweets, non-standard abbreviations and slang used referring to locations, ambiguities in identifying exactly a location. Indeed, “the geoparsing problem intensifies when the language is informal, or contains non-grammatical microtext that may contain uncommon abbreviations or slang” \cite{78}. In particular, two types of ambiguities are relevant \cite{4}: geo/non-geo ambiguities, which happen when a location name is also a proper non-geographic name or a common name\footnote{In this case the problem is worsened with respect to a traditional setting by the fact that often social-media text is bad-capitalized.} and geo/geo ambiguities, which happen when two or more locations share the same name. For example “Turkey”/“turkey” is a geo/non-geo ambiguity and “London, England”/“London, Ontario” is a geo/geo ambiguity. A gazetteer like GeoNames provides alternative names for locations that can be used to increase the coverage \cite{33}. Also more specific and invariant elements of the message text have been used to get more coarse-grained insights or to evaluate trends: the URL links, hashtags, and the used words themselves.

\cite{21} focuses on the locations mentioned in tweets during natural disasters. A state-of-the-art library to perform named entity recognition in a traditional setting (Stanford NER) is trained to find locations in tweets, after having manually annotated a gold standard. The results highlight how such a library is not able to deal with bad capitalization, misspellings etc., that are plentiful in microtext.

\cite{55} is an example of statistical technique to perform toponym resolution: they trained a log-linear model with 2 features to solve both geo/geo and geo/non-geo ambiguities and they achieved a precision of 15.8\%, where the baseline obtained by simply checking against the gazetteer all the noun phrases achieved a precision of 4.93\%. A similar approach is shown in \cite{44}, where they used different machine learning algorithms for the disambiguation task: they trained classifiers using the candidate locations retrieved in GeoNames as features and the state as target class to disambiguate.
In [44] a comparative analysis of geoparsing tools on social media is made. They manually annotated a corpus of social media sentences and they tested several geoparsing/NER tool, including both tools for traditional text and tools specifically tuned over social media. The results show that in general off-the-shelf NER systems achieve poor results and a system designed for traditional text (Stanford NER) in certain cases outperforms a system specifically designed to analyze twitters (TwitterNLP).

[33] focuses in detecting and disambiguating locations mentioned in Twitter messages, facing both geo/non-geo and geo/geo ambiguities. Geo/non-geo ambiguities are addressed as a NER task limited to locations named entities, geo/geo ambiguities are addressed through several heuristic disambiguation rules. For the NER task a dataset is manually annotated and Conditional Random Fields modes (CRF) are trained with various sets of features. Four features are considered: bag-of-words (unigram), part-of-speech tags (POS), adjacent tokens and POS tags and whether the token appears in the gazetteer or not (which is a necessary but not sufficient condition to being a location). These features are tested also separately to compare their contribution in the result, and a naïve approach that just search in the gazetteer is used as baseline. The POS tagger of TweetNLP [54] is used. The classifier, in addition to detect locations, assigns a location type (e.g. city, country, etc.) to each one. The results show that in general identifying cities is more difficult than identifying countries or state/provinces since the total number (hence the possible ambiguities) is much greater. Focusing on cities, the baseline approach has a very low F-score\(^{11}\approx 0.23\) caused by the very low precision; using only the bag-of-words feature the performance increases with F-measure \(\approx 0.70\) and the best precision, while using all the features an F-measure \(\approx 0.82\) is reached. An error analysis highlights incomplete/uneven coverage in the training data and misspelled locations as sources of errors. Geo/geo ambiguities are addressed trying a series of heuristics until the first one returns a unique location: first of all the possible matches from the gazetteer are filtered keeping only those with the same location type as the type detected by the classifier, then adjacent locations are searched for hierarchies like “Los Angeles, California, USA”, then if other locations are already matched in the same tweet the nearest location is selected and at the end if none of the previous steps can decide a unique location, the most populated one is returned. The results show that using adjacent locations brings better results, while using the distances to the other locations in the same tweet lowers the accuracy.

\(^{11}\)Harmonic mean of precision and recall. These measures are formally defined in Section 4.3.
uses supervised machine learning to weigh the different fields of the Twitter message and the features of the gazetteer (GeoNames) to create a model that will prefer the correct gazetteer candidate to resolve an extracted expression. They train support vector machines (SVM) using different sets of features: gazetteer features, which include population, alternative names and level in the geopolitical hierarchy and act as baseline; contextual features, which take into account other toponyms in the same tweet under the assumption that “each location mentioned in a particular context is not independent, but rather is spatially correlated” using both “hierarchical” and “sibling” relationships so that co-occurring toponyms help disambiguating one another; metadata features, which include the user’s location if it is correlated to the toponym and the place field (however, they had to discard the time zone because it decreased the accuracy); word form features, which account for the morphological similarity between the extracted toponym and the gazetteer candidates.

In location recognition and linking are performed together in a joint search space, differently from the traditional pipeline setting. The focus is fine-grained locations (like restaurants, landmarks, shops, etc.) in tweets. They formulate the end-to-end location linking problem as a structured prediction problem and propose a beam-search based algorithm. This allows to overcome two limitations of the traditional recognition-linking pipeline: the fact that all the errors in the first step are propagated to the second one without feedback and the fact that there are cross-dependencies among the two tasks which are not modeled in the pipeline architecture. They also propose a semi-supervised learning algorithm to improve the results using unlabeled data.

Statistical techniques are also used for the inference task. In the user’s city-level location is inferred using a classifier to correlate cities and words in tweets. Their system is able to place 51% of Twitter users within 100 miles of their actual location. In an ensemble of several classifiers is used to predict the locations of Twitter users from the tweet contents (unigrams, hashtags and place names), tweeting behavior (volume of tweets per time unit) and external knowledge (gazetteer), in combination with several heuristics. In this case the focus is home locations of users at different granularities. is another example this kind of technique.

In the location at the finest level of granularity for tweets in the context of emergence response is found using several location-related sources: textual content,

\[\text{Structured prediction means using supervised machine learning techniques to predict structured objects, in this case the outcome of the recognition and linking tasks.}\]
user profile location and place labels. The location at the finest granularity from the potential sources is assigned to each tweet. The method was able to successfully infer the location of 87% of tweets with an average distance error of 12.2 km and the median distance error of 4.5 km.

### 2.4.2 Network-based methods

Network-based methods exploit the property that users on social media tend to interact mostly with the same people they interact in their lives, so this property is used to individuate the location of a user (or a tweet) based on the locations of his “friends”. Indeed, “In many cases, a person’s social network is sufficient to reveal their location” [61]. Since most of the network-based methods in the literature focus on relationships among users, they usually target user home residence.

In [16] following-follower relationships on Twitter are used to infer the locations of users. They used a simple method which take into account the most recurrent location among friends. With such a method, they demonstrated that users with too few friends cannot be geolocated easily because their friends are a weak evidence, while users with too many friends cannot be geolocated easily because their friends are too sparse. Instead, users with an intermediate number of friends provide better information for location inference and they reported that their method can improve the amount of approximately locatable users by up to 45%, with reasonable confidence.

In [47] a decision tree is trained to show that some features of relationships (for example users who mention each other) are correlated with physical proximity.

Also [61] proposes a method to infer home locations of users in Twitter based only on their social networks. They formulated geolocation as a classification task where the class is the inferred location. They trained support vector machines (SVM) classifier to assign the location of a user based on his friends with known locations. As features they used: the number of occurrences of each location among the friends, features related to the population size of those locations (a novelty of their work is that they account explicitly for the density of people in an area), features to model relationships that form triads and reciprocal relationships. Their method is able to accurately predict the location (at the city/town/village/hamlet granularity) in the 50% of the cases, while simply choosing the most common location among friends gives the correct result in the 39% of the cases and a

---

13 Depending on the context and the social media, this relationship could have various meanings, from a “following” relationship to an actual online “friendship”.

---

27
2. State of the art

random assignment gives an accuracy of 31%.

[36] uses label propagation (actually, an extension which interprets labels spatially) through the social network to assign to nearly all the users a location, starting from few labeled users. Both attached coordinates and self-reported locations are used, highlighting a small loss in precision. They reported an estimated median error under 10 km.

[22] proposes an algorithm which addresses both the disambiguation and the inference problems in social media, focusing on the user’s locations. Their algorithm chooses the right interpretation of a self-reported toponym of a user by using the (possibly ambiguous) self-reported toponyms of his friends as a context. In particular, it builds a location graph where self-reported toponyms are connected when they are related, that is when are part of the administrative division, so that the weight of a link between two self-reported toponyms measures the granularity of the administrative division shared by them. Then they determine the most important interpretation based on the assumption that more important interpretations are likely to receive more links from the interpretations of the self-reported toponyms of the user’s friends. As for the inference problem, they apply the same strategy relying solely on the interpretations of the user’s friends.

[58] directly compares a text-based method based on classification to a network-based method. Moreover, a hybrid method for location inference is proposed, which is both text-based and network-based. On one side, they use a classifier to infer the location of users based on their tweets using unigrams as feature, on the other side they use label propagation like in [36] connecting users based on @mentions. Overall, the hybrid method outperforms the two baselines. They highlight that the network-based method is more robust than the text-based one (less hyper-parameters, less used resources, converges faster and geolocates also users which are not in the test set) but it fails in geolocating disconnected users, therefore the relative performance depends on how much the dataset is connected.

Notice that the majority of works in this field focus only on the English language, like [14, 33, 40, 44, 58, 61, 78], even if generally speaking most of them would work in the same way with other languages. In particular, methods based social networks are completely language-independent in each phase [5], while methods based on classification would require a training set which accounts for all the target languages. The same holds for methods which rely on natural language processing algorithms, which must be pre-trained on the target languages. A possible limitation in a multi-language environment is given by the external knowledge (the gazetteer), which could not cover or could cover unequally or with a different qual-
ity some regions of the world. This is an unavoidable factor in a geocoding task, since “the output of any geocoding algorithm is only as exact as the knowledge base that underlies it” [78].

Many of the cited papers focus on the users geolocation, not tweets geolocation (e.g. [22, 36, 17, 58, 61]) or geolocate tweets through users geolocation [16]. Indeed, the location of users is the focus of the majority of the network-based algorithms. This is due to the fact they they focus on explicit (articulated) networks, like friendship, as opposed to implicit (behavioral) networks, which are inferred from communication patterns. Explicit networks model “static” relationships among users, therefore they do not capture entirely the relationships among actual messages posted, while “implicit networks may better reflect the online activities of a person” [13]. Instead, the new algorithm for geolocation proposed in Chapter 3 aims to find the locations of tweets (and posted contents in general), and it achieves it focusing on implicit networks (like replies and mentions). Indeed, “implicit networks are particularly interesting in the crisis scenario because many exchanges happen among people who were not connected before the crisis” [13]. Nonetheless, it is based on a network to achieve geolocation similarly to the papers described here.

2.5 Dimensions and open problems

In this section are summarized the main dimensions of this thesis, focusing in particular on the open problems addressed.

The main dimensions are as follows.

- **Content-based.** In the context of geolocation it means message-based geolocation as opposed to, for example, user-based geolocation. This choice allows to use geolocation for content-based analysis: document-based event detection (ref. Section 2.2) and many applications like image analysis that will be detailed in Chapter 4.

- **Stream processing.** Data should be processed as soon as are available without any prior knowledge about them. In the context of geolocation, in particular network-based geolocation (ref. Subsection 2.4.2) this means working on an evolving network. Most of the cited papers, instead, focus on static networks completely available from the beginning. In the context of emergency management in general (and therefore the applications presented in Chapter 4)}
2. State of the art

this means not having prior knowledge about the events and their locations, but having only a partial and incremental view as they unfold.

- **Online** processing. In this context it means being able to processes new elements in a time independent from the number of already processed elements (on average). This property, together with stream processing, is a prerequisite for real-time processing, which means setting precise latency limits and also depends on the architecture and the implementation (real-time processing is out of scope in this thesis). This a prerequisite for a scalable system.

- **Domain-specific.** Not only analyzing social media is very different from analyzing traditional text [60], but also crisis situations present unique challenges [29]. Therefore, the unique properties of social media and crisis situations should be exploited to overcome the related challenges. In the context of location disambiguation on social media, even if often the goal is to support emergency management, most of the cited papers are not tailored to that specific domain. Domain-specific approaches have been proposed for content analysis and event detection, and will constitute the applications presented in Chapter 4. Notice that being domain-specific does not imply that the proposed techniques cannot be applied to other different domains; indeed most of them have been applied or could be applied also to different domains with similar properties, both in the social media context and in other contexts.

- **Language-independent.** Social media are used all around the world and many places where crisis situations are frequent, especially natural disasters, are not English-speaking countries. An approach to account for all the different languages is being language-independent, that is not relying on language-specific features. However, in the context of location disambiguation there are intrinsically dependencies on the language. Therefore, the goal will be to not add additional dependencies and not rely on language-specific features when possible.

---

14For example gazetteers do not cover all the countries equally.
Chapter 3

Geolocation

In this section a new algorithm for geolocation on social media is proposed.

The algorithm disambiguates the toponyms cited in text messages (like tweets) to find the locations related to their content, and can be also used to infer the locations when no toponyms are mentioned explicitly. This is useful to associate a location to any information, metadata or multimedia extracted from tweets.

The algorithm is based on the idea of collaborative geolocation, that is exploiting the natural connections that are created by users during an event on a social media to add a context to each single message, overcoming in this way the problem of noisy and decontextualized messages typically found on social media.

It is specifically designed to efficiently geolocate social media messages during a precisely located event, like a natural disaster and a crisis scenario in general. This specificity, which prevents it to be considered a general purpose geolocation algorithm, allows it to exploit the features of this particular context maximizing the effectiveness.

It is a graph-based framework that can be implemented with many variants and different metrics; the current implementation is detailed alongside the explanation of the algorithm in general and more implementation details are at the end of the chapter.

The current implementation works on tweets, even if the algorithm itself does not pose any restriction on the social media. For this reason, in the following the algorithm will be described using a Twitter-based terminology (e.g. “tweets” rather than “messages”).

In the following, with “location” referred to a tweet, the set of locations associated to the tweet itself is meant.

The principles behind the algorithm are as follows:
• Each tweet has an a priori “local” set of toponyms, that can be fully disambiguated, ambiguous or simply missing. This depends only on the tweet itself, that is on its local context, which is built starting from the toponyms mentioned in the single tweet and looking for mutual relationships.

• Each tweet, as soon as it is inserted into the system, is connected to other related tweets, and then new related tweets will be connected to it, through an online graph-based algorithm. The concept of “relationship” or (“similarity”, from a formal point of view) here is very broad; each tweet should be connected to related tweets both from a temporal, semantic and social point of view, privileging dynamic links over static links, that is behavioral relationships over explicit relationships [13].

• A tweet connected to other disambiguated tweets finds in them a possible global context. The global context can allow the disambiguation of the ambiguous toponyms of a tweet and the inference in case of missing toponyms.

• An ambiguous tweet, as soon as it is disambiguated, can provide a global context to its neighbors — and this happens recursively. This means that an ambiguous tweet after its disambiguation not only provides a location for the event, but becomes itself a helpful source to disambiguate and infer other tweets.

Following this approach, the algorithm is able to resolve both geo/geo and geo/non-geo ambiguities, thanks to the local context as first attempt and the global context as further attempts. As soon as a tweet finds a context useful for its ambiguous toponyms, they are disambiguated.

A location in a tweet is disambiguated only when it has a context, and this reduces substantially ambiguities because usually for \( N \) candidate locations only the “real one” matches the context (for geo/geo ambiguities) and no one matches the context in case of a false positive in a geo/non-geo ambiguity.

Therefore, the algorithm is characterized by the following advantages, which lay the foundation on well-known features of social media:

• The use of each location to disambiguate the others, in a geographical collaborative way. If both locations \( L_1 \) and \( L_2 \) are ambiguous, they can mutually help their disambiguation. Indeed, “each location mentioned in a particular context is not independent, but rather is spatially correlated” [78].
• The use of each tweet to disambiguate the others. Twitter is both an information network and a social network \[13, 49\] and the algorithm takes advantage of the relationships among tweets and users to improve the geolocation for each single tweet.

• The propagation of reliable information, since only tweets already disambiguated can propagate their context, but not uncertain information, since ambiguous tweets do not propagate anything. Since Twitter is a noisy environment \[56\], this allows to limit the impact of noise cutting down its propagation and privileging reliable information instead.

The algorithm is designed to work *incrementally* and *online* and receives the stream of tweets to process.

Moreover, the algorithm is designed to be completely *language-independent*, because it does not rely on any language-specific feature. This allows its application even for resource-poor languages.

This chapter is organized as follows. In Section 3.1 an overview of the algorithm is outlined, alongside with a high-level description of the algorithm. Before starting a detailed description of the algorithm from Section 3.3, it is exemplified in Section 3.2 to show the main features in action. The chapter is concluded by a discussion in Section 3.7, which focuses on optimizations, limitations, and comparisons with other techniques.

### 3.1 Algorithm Overview

The algorithm uses a *graph of graphs* as data structure.

At each point in time, each tweet has a *state* among: AMB (ambiguous), MIS (missing location), DIS (disambiguated) and INF (inferred). With these acronyms also the respective set of tweets will be indicated in the following.

Each tweet has a set of *candidate locations*, which are the ambiguous locations (related to the toponyms in the tweet) to disambiguate. If this set is empty, the location of the tweet must be inferred instead. Therefore, AMB and MIS are *start states*, depending if some candidate locations exist or not (the overall state diagram is summarized at the end of the chapter, Subsection 3.7.3). The candidate locations come from the *toponyms* in the tweet, which are obtained in a previous toponym recognition phase\(^1\).

\(^1\)The algorithm could work also considering as candidate locations all the n-grams in the text,
3. Geolocation

As a new tweet arrives, it is translated into a local graph, which accounts for relationships among candidate locations in the tweet, so that each candidate location can disambiguate the others providing a context to them. This is the local disambiguation phase and at the end, if some relationships have been found, the involved locations are disambiguated and the the tweet itself is disambiguated. If no relationships have been found but there exist candidate locations the tweet is considered ambiguous, if simply there are no candidates the tweet is marked as missing location.

Moreover, each tweet is inserted into a global graph which accounts for relationships among tweets. This provides to ambiguous or missing locations tweets a global context. The global context is built starting from the local graphs of the disambiguated neighbors. Thanks to the global context, an ambiguous tweet has a new chance to be disambiguated, while the location of a missing location tweet can be inferred.

Notice the difference among local graphs and the global graph. Each local graph is a graph of toponyms mentioned in the individual tweet, and two toponyms are connected if there is a geographic relationship among them (and in this case they are directly recognized and disambiguated as locations); the local graph is the local context of each tweet. The global graph instead is a graph of tweets and since each tweet has its own local graph, it is actually a graph of graphs, and two tweets are connected if there exists a relationship among them (see Section 3.4). Thanks to the global graph, each tweet can have some neighbors and can use the local graphs of its neighbors to build its global context, which is a graph of the toponyms disambiguated in the neighbors (with their mutual relationships) that can provide a context for the ambiguous toponyms of the tweet (or to infer the location, if missing).

The algorithm employs a sliding window model to update the global graph: this allows accounting for context changes, privileging dynamic links and prevent any memory limitation problem.

The algorithm is designed to be online. In this context, this means that each new element (i.e., tweet) can be processed on average in a time independent from the number of tweets already in the system. This property has to do with the ability of the algorithm to scale well.

therefore performing itself the toponym recognition phase as a joint task, but it is impractical. In any case, the algorithm does not take the toponyms given as input blindly as correct, but considers them actually locations only if able to disambiguate them, performing therefore a “refinement” over the previous toponym recognition. This discussion is detailed in Section 3.3.
The “main cycle” is described in Algorithm 1. Its role is to keep updating the global graph as soon as new tweets arrive, handling the sliding window. For each new tweet, the function `new_tweet` is called (row 6), which is described in Algorithm 2. The role of this function is to build the local graph, so that a tweet can eventually be disambiguated locally, and assigning to each tweet its starting state among `AMB`, `DIS` and `MIS`. A disambiguated tweet call the function `action` on all its (ambiguous and missing location) neighbors, while an ambiguous or missing location tweet call the function `action` on itself. This function, which is described in Algorithm 3, constitute an “attempt” to globally disambiguate (in the ambiguous case) or to infer (in the missing location case) a tweet. Therefore, the global graph is built and is used to make the attempt which could bring to the state transitions `AMB → DIS` and `MIS → INF`. In case an ambiguous tweet is successfully disambiguated, it will try to disambiguate/infer its neighbors in a recursive manner.

Notice: all the listings described in this chapter in pseudo-code are high-level simplifications. Some features of the algorithm are omitted to keep them readable, like some variables of secondary importance (which, however, are described in the next sections) and data structures managed for performance reasons. Moreover, there is a not rigorous use of attributes and global functions and the function parameters which are not essentials or would be too complex are omitted or simplified. The optimizations in the current implementation, described in Subsection 3.7.1 are not reported in the listings because would make them less readable.

A rigorous and mathematically modeled description of the algorithm is available in the following sections.

---

**Algorithm 1 main**

1: `while tweet = get_next_tweet() do`  
2: `update_sliding_window()`  
3: `neighbors ← compute_most_similar(tweet)`  
4: `global_graph.nodes ← global_graph.nodes ∪ tweet`  
5: `global_graph.edges ← global_graph.edges ∪ \{(tweet, i) \mid i ∈ neighbors\}`  
6: `new_tweet(tweet)`  
7: `end while`
Algorithm 2 new_tweet

Input: tweet
1: surface_forms ← S(tweet)
2: candidate_locations ← \{U(s) | s ∈ surface_forms\}
3: local_graph ← make_local_graph(candidate_locations)
4: if count(local_graph.edges) ≠ 0 then
5: state ← DIS
6: location ← finest_granularity(local_graph)
7: for each neighbor in neighbors(tweet) do
8: if neighbor.state ≠ DIS and neighbor.state ≠ INF then
9: action(neighbor)
10: end if
11: end for
12: else if count(local_graph.edges) = 0 and count(candidate_locations) ≠ 0 then
13: state ← AMB
14: action(tweet)
15: else
16: state ← MIS
17: action(tweet)
18: end if
Algorithm 3 action

Input: tweet
1: disambiguated_neighbors ← \{n ∈ neighbors(tweet) | n.state = DIS\}
2: if count(disambiguated_neighbors) = 0 then
3:  return
4: end if
5: global_graph ← make_global_graph(disambiguated_neighbors)
6: if tweet.state = AMB then
7:  location ← disambiguate_global(global_graph, tweet.candidate_locations)
8:  if location is not null then
9:    tweet.state ← DIS
10:   tweet.local_graph ← derive_local_graph()
11:  for each neighbor in neighbors(tweet) do
12:    if neighbor.state ≠ DIS and neighbor.state ≠ INF then
13:      action(neighbor)
14:  end if
15: end for
16: else if not_disambiguable(tweet) then
17:   tweet.state ← MIS
18: end if
19: else if tweet.state = MIS then
20:   location ← infer_location(global_graph)
21:   if location is not null then
22:     state ← INF
23: end if
24: end if
3.2 Example

Before describing the algorithm in details starting from the next section, an example to understand its main concepts is sketched in the following.

This example does not use real tweets and real locations, but only aims to show a typical execution. Examples based on real tweets and locations are available in Chapter 5. Moreover, this example does not pretend to include all the features described in the following but only to give insights about the main ones.

Suppose there is a tweet which refers to a location A posting a photo, as shown in Figure 3.1. That *toponym* (or *surface form*) is ambiguous because many places around the world are called “A”, and, moreover, A could not be a toponym at all but simply a common word mismatched as toponym in the toponym recognition phase. The toponym A is related to a set of *candidate locations*, that is the set of locations around the world actually referred to as “A”. The algorithm tries to build a *local context* trying to find relationships among the candidate locations related to the toponyms mentioned in the tweet’s text. In this case there is only one toponym A and no relationships can exist, therefore the local context (shown in the rectangle associated to the tweet) is made only by one *decontextualized* node, A. Since no locations have been disambiguated using the local context, because it contains only isolated nodes without relationships (that is, all the toponyms are still *ambiguous*), the tweet itself is considered *ambiguous*. At this point the algorithm tries to find *neighbors* of the tweet, that is *related tweets* which can be connected in the *global graph*. In this case no related tweets are found and therefore, since the ambiguous tweet has no disambiguated neighbors (and no neighbors in general), no further attempts to disambiguate its toponyms using a global context can be made and the algorithm stops waiting for new tweets.

A new tweet arrives, as shown in Figure 3.2. In this case the algorithm is able to find relationships among the candidate locations related to its mentioned toponyms, that is finding among all the locations around the world named “B”, “C” and “D” (which constitute the set of candidate locations) exactly those locations which have a relationship with each other: it finds two locations named B and D which are contained in a location named C (for example C could be the name of a city and B/D could be two districts of C). In this case, B, D, and C have been *disambiguated* since they are connected to other locations in the local graph (shown in the rectangle below the tweet). The tweet itself, since has at least a disambiguated location, is considered disambiguated: this is an example

---

2 The method used to find related tweets is detailed in Section 3.4.
of local disambiguation. At this point the algorithm tries to find neighbors of the
tweet, which could benefit from this disambiguated tweet for their own ambiguous
toponyms, but no related tweets are found. The algorithm stops waiting for new
tweets.

A new tweet arrives, as shown in Figure 3.3. It refers to the location A, which,
like the first analyzed tweet, cannot be disambiguated locally. Therefore also
this tweet is labeled as ambiguous. However, it is considered related to the first
two tweets received (notice that it shares some words with both) and therefore is
connected to them in the global graph, as shown in Figure 3.4.
3. Geolocation

Since among its neighbors there is a disambiguated tweet (the second one), a global disambiguation attempt can be made. All the local graphs of its disambiguated neighbors are joined (taking into account also the new relationships among their contained locations) to create the global context for the node. In this case, since there is only one disambiguated neighbor, the global context of the third tweet coincides with the local context of the second tweet, as shown in Figure 3.4. The ambiguous toponym A now “sees” the context composed by the disambiguated locations B, D and C, and, among all the candidate locations around the world referred to as “A”, it is possible to find a location named A which is part of B, disambiguating it. Now the local context of the third tweet not only contains the disambiguated location A, but also B and C because they are part of its hierarchy (but not D, which is not). This is an example of global disambiguation.

The third tweet, as soon as it has been disambiguated, in turn tries to disambiguate its ambiguous neighbors (that is, the first tweet). Also in this case the global context seen by the first tweet is generated by only one node, as shown in figure 3.5, and coincides with the local context of the third tweet. Among the candidate locations related to the toponym A in the first tweet, one location A is in the global context, therefore A is directly disambiguated and the local context of the first tweet is composed by the new disambiguated location and its hierarchy (B, C). Recursively, the first tweet tries to disambiguate its neighbors, but it has no ambiguous neighbors therefore the algorithm stops, waiting for new tweets.

Notice: in the diagrams a toponym in yellow is an actual location (that is, a disambiguated toponym), instead a toponym in white is a candidate location (that is, the set of all the possible locations the toponym could refer to).
3.2. Example

Figure 3.4: Example/4.

Figure 3.5: Example/5.
3. Geolocation

3.3 Local disambiguation

The input to the local disambiguation phase is a set of surface forms (or toponyms). A surface form (or anchor phrase) is “a textual phrase which can potentially link to some entities” \[24\]. In this context of location entities only, a surface form is a set of one or more consecutive words which can potentially refer a specific location in the external knowledge. As already mentioned in Section \[2.4\] there exist a many-to-many relationship between surface forms and locations in the external knowledge, because each location can have different alternate names besides its official name and can be cited only partially in the tweet, and different locations can share the same names and/or alternate names.

From a theoretical point of view is not necessary to use an external named entity recognition (NER) library to extract a set of surface forms from a tweet to use this algorithm effectively, since the algorithm itself has geo/non-geo disambiguating capabilities (that is, to some extent, entity recognition capabilities), besides geo/geo disambiguating (that is, entity linking capabilities). However, this would require to test each possible \(n\)-gram of the tweet as a surface form, that is it would be equivalent to use a “dummy” NER module which simply returns each possible \(n\)-gram. Therefore, for performance reasons is useful to start with a set of surface forms provided by an external library, possibly privileging the recall since the algorithm is able anyway to solve geo/non-geo ambiguities and correct it, while a low precision of the NER library cannot be corrected by the algorithm because it does not consider any surface form except those provided as input.

The local disambiguation phase takes into account the location names in the single tweet, trying to build a local context, or, equivalently, trying to find some locations that match the surface forms in the tweet which are in relationship with each other, so that each one can disambiguate the others.

If such relationships exist, each location in the relationship can be considered reliable and the most fine-grained ones are took as locations of the tweet itself.

In general, any relationship between the locations could be employed. In the developed system, a hierarchical relationship has been chosen, so that a location \(l_1\) is in relation with a location \(l_2\) if \(l_1\) is directly included in \(l_2\) from a geographical perspective. For example, Brooklyn is included in New York City. In this case, \(l_2\) is the parent location and \(l_1\) is the child location.

This kind of relationship has been chosen since it is a strong relationship (for each location typically there is only one or a few locations that directly include it), it is parameter-free (differently, for example, from a relationship based on distance).
and it is efficient to be computed\footnote{Indeed, it is just a check in a table where all the pairs in relation are stored. This check can be computed efficiently choosing the child location as \textit{index}.} It has been successfully used in the literature for geolocation \cite{22,33}.

This relationship is a \textit{partial order} (like inclusion) and the locations can be represented as a \textit{directed graph} (conventionally, an edge goes from a child location to a parent location).

If at least a relationship is found, the resulting graph will have at least two nodes. Each node of the graph is a disambiguated location, the tweet itself can be considered disambiguated and the graph as a whole will provide a context for the “near” (that is, related) tweets in the global disambiguation phase. If there are candidate locations, that is surface forms that potentially could link to some locations in the external knowledge, but is not possible to find local relationships, they are considered ambiguous and the global context will be used. If there are no candidate locations, that is there are no surface forms or each surface form link to no locations in the external knowledge, the location for the tweet is considered missing and the the only possibility will be inferring it based on the global context.

At the end of the local disambiguation phase, the tweet can be in 3 different states:

\textbf{Disambiguated} with a reliable location and its context

\textbf{Ambiguous} with a list of candidate locations without context

\textbf{Missing location} without any candidate location

In the following, the local disambiguation phase is detailed and formalized.

\textbf{Definition 3.3.1 (Location).} A location $l$ is a non-ambiguous entry in the external knowledge base (e.g. GeoNames), identified by a unique ID. It has one or possibly more names which allow referring to it.

\textbf{Definition 3.3.2 (Surface form).} A surface form (or anchor phrase) $s$ is a textual phrase which can potentially link to some locations.

\textbf{Definition 3.3.3 (Candidate location).} A candidate location is a potential location a surface form could link to.

\textbf{Surface forms of a tweet} Each tweet $t$ has a set $S(t) = \{s_1, s_2, \ldots\}$ of surface forms, possibly empty.
3. Geolocation

**Candidate generation** The (possibly empty) set of candidate locations a surface form could link to is obtained through a search function $U$:

$$U(s) = \{l_1, l_2, \ldots\}$$

The function searches for (possibly partial) matches of the surface form with the location names in the external knowledge.

Given the set $S(t) = \{s_1, s_2, \ldots\}$ of surface forms, it is possible to obtain the complete set of candidate locations for the tweet as

$$C(t) = \{U(s_1), U(s_2), \ldots, U(s_N)\} = \{C_1, C_2, \ldots, C_N\}$$

**Location relationships** Given $C(t)$, there are $\binom{N}{2}$ combinations of candidate locations sets that must be searched for a relationship. For each pair of candidate locations sets $C_i$ and $C_j$ with $i \neq j$ such that $|C_i| = p$ and $|C_j| = q$, $p \times q$ checks are necessary.

Notice that, even if the number of checks grows as a combination of all possible candidate locations for all the different surface forms, the short nature of tweets, the scarcity of mentioned locations and the very efficient relationship checks ensure that this phase is not a bottleneck in practice. Anyway, since each tweet has a fixed size, this step can always be done in a bounded time satisfying the online constraint.

As said previously any kind of location relationship $R$ could be employed and in the current implementation the hierarchical relationship $H(\text{parent}, \text{child})$ has been chosen. This means that, given $C_i, C_j \in C(t)$ with $i \neq j$, there exists a relationship between $l_{ID_i} \in C_i$ and $l_{ID_j} \in C_j$ if $H(l_{ID_1}, l_{ID_2})$ or $H(l_{ID_2}, l_{ID_1})$.

Moreover, any kind of indirect relationship could be tested next to the simple membership to the $H$ relationship to increase the number of matches. In the current implementation, it has been chosen to additionally check for a “child of the same parent” relationship, that is checking if

$$\exists l_{ID_3} \mid H(l_{ID_3}, l_{ID_1}) \land H(l_{ID_3}, l_{ID_2})$$

With $l_{ID_1} \in C_i$, $l_{ID_2} \in C_j$, $C_i, C_j \in C(t)$ and $i \neq j$. In this case, $l_{ID_3}$ is included among the disambiguated locations even if does not exist any $s$ such that $U(s) = l_{ID_3}$.
This kind of check is parameter-free and can be computed efficiently.\footnote{Having a relation \((\text{parent-id}, \text{child-id})\) with an index on \((\text{child-id})\), performing this check requires only 2 direct accesses, which are not additional with respect to computing only the direct parent-child relationships.}

At the end, a set of parent-child pairs \(P(t) = \{(l_1P, l_1C), (l_2P, l_2C), \ldots\}\) is returned. \(\text{locs}(t)\) is defined as the union of all the locations in \(P(t)\).

The following holds:

- If \(P(t) \neq \emptyset\) the tweet has been \textit{disambiguated}
- If \(P(t) = \emptyset\) and \(\mathcal{C}(t) \neq \emptyset\) the tweet is \textit{ambiguous}
- If \(\mathcal{C}(t) = \emptyset\) the tweet is \textit{missing location}

**Local graph generation**  After all the relationships have been identified, for a disambiguated tweet it is possible to generate the local graph \(g(t) = (\text{locs}(t), P(t))\). It is a directed graph with an edge for each member of \(P(t)\).

Each location \(l \in \text{locs}\) is a \textit{disambiguated location}, and \(g(t)\) is the (local) context led by \(t\).

In case a tweet can be disambiguated locally, the locations of the tweet are chosen as those with the finest granularity in \(\text{locs}\). This reflects the assumption that if both \(l_1\) and \(l_2\) are cited in a tweet and \(l_2\) is a part of \(l_1\), then the focus is primarily on \(l_2\) and \(l_1\) is mentioned only for context or to frame the specific sub-event in its entirety. The following definition formalizes this:

**Locations for a locally disambiguated tweet**  Given a locally-disambiguated tweet \(t\), its locations are

\[
L(t) = \{l_i \mid \exists (l_j, l_i) \in P(t) \land \nexists (l_i, l_k) \in P(t)\}
\]

Notice that, even if the tweet has been disambiguated, there could be candidate locations sets with no disambiguated candidates, i.e. there could exist some \(C_i \in \mathcal{C} \mid \forall c \in C_i, c \notin \text{locs}(t)\), because the only requirement to have a disambiguated tweet is being able to disambiguate \textit{at least one} surface form. An objection could be that the tweet has been only \textit{partially} disambiguated, since \(C_i\) could in fact contain a meaningful location that cannot be disambiguated for a lack in the local context. This problem, in theory, could be overcome changing the definition of disambiguated tweet such that it requires that each \(C_i \in \mathcal{C}\) must bring to a resolved location. However, this solution does not take into account the fact that...
a surface form $s$ could be a false positive (that is, not meant to link a location, even if accidentally $U(s) \neq \emptyset$, like in case of geo/non-geo ambiguities) or that the external knowledge does not contain the location meant in the tweet, even if it is a real one: in these cases, no candidate locations in $C_i$ should be disambiguated.

Therefore, near to the new definition of disambiguated tweet, a method to decide if a candidate location certainly does not match would be required, and it should be guaranteed that after a period of time each candidate location is recognized either as matching or not matching, to give in output a tweet with a precise state. Besides the difficulty in designing such a method, a challenge would be to satisfy the online constraint of the algorithm. For these reasons, it has been chosen to consider all the locations that is possible to disambiguate at first as the locations of the tweet, avoiding the state of “partially disambiguated” tweet. Solutions to this could be further investigated in future work. For the global disambiguation phase, described in Section 3.5, the same considerations apply.

### 3.4 Global graph management

Once a tweet is inserted into the system, it is connected to other tweets in a global graph $G = (V, E)$.

The graph $G$ includes all tweets in the last period of time $T$, using the sliding window technique. This allows accounting for context changes and preventing the memory limitation problem.

As mentioned previously, the algorithm itself does not focus on the link between tweets but rather on how to use them for the geolocation task. The assumption is that these links connect tweets which have a relationship in the present (for example in the context of the current emergency), that is implicit relationships.

Indeed, social media are characterized by two kinds of social networks: explicit (articulated) social networks and implicit (behavioral) social networks. Explicit social networks are based on codified relationships, like the “friendship” or the “following” relationships, while implicit social networks are inferred from communication patterns. As reported in [13], “implicit networks are particularly interesting in the crisis scenario because many exchanges happen among people who were not connected before the crisis”.

\[5\] Indeed, in the hypothesis of reducing the problem to only geo/geo ambiguities and assuming a “perfect gazetteer”, finding the correct candidate would be only matter of time (provided that the useful context sooner or later is linked).
In the current implementation, the links are created based on two implicit relationships:

**Tweet similarity** measured as the cosine similarity among the term frequency-inverse document frequency (tf-idf) vectors. For the hashtags just the term frequency is used to increase their impact given their natural semantic role of grouping tweets. Two tweets are considered similar, and therefore linked, if the similarity is greater than a fixed threshold, which is a parameter. This kind of similarity is well tested to group together tweets related to the same event \([9, 10, 59]\) and models a relationship given by two users talking about the same topic at the same time (since two tweets must be together in the sliding window to be considered similar).

**User relationships** exploiting mentions in tweets. A user can mention another user in a tweet, establishing a relationship. Such relationships are implicit and strictly related to the present time, since a relationship exists only if a user \(A\) has mentioned a user \(B\) in the current sliding window, independently from the fact that the two users are friends or not. This is a user relationship managed through a user-based support graph; the graph \(G\) is a tweet graph therefore two tweets are connected if their respective authors are in relationship. Each author is considered in relationship with himself, so that his recent past messages\(^6\) provide a context for his future messages.

The algorithm does not make differences among the different kinds of links (that is, \(G\) is not weighted). The relationships used in the current implementation are chosen because successfully used in the literature for similar tasks. Investigating the impact of different relationship types on the geolocation performance, as well as weighted links in the graph \(G\) could be the subject of future work. Indeed, as explained in the introduction, the proposed algorithm is a framework that can be implemented with many variants and different metrics and the main variants are linked to the global graph management.

Even if in principle the algorithm is independent from the management of the global graph and the only requisite is that the global graph models behavioral relationships among messages, in practice the performance will vary choosing different kinds of relationships and setting different thresholds, and also the time required to process a new tweet is function of the types of relationships chosen. A naïve

\(^6\)That is, those in the sliding window.
approach that compares each tweet with all the others is not scalable and approximations should be introduced, for example based on hashing like [56]. However, approximations themselves introduce new parameters. Moreover, also the relationships based on mentions could be different: for example it is possible to connect all the messages of two users when one of the two mentions the other, connect only the messages constituting the discussion or consider only mutual mentions.

Even if these relationships have a big impact on the performance of the algorithm, their study is out-of-scope in this thesis. There is a line of research totally focused in studying the social networks [49] and evaluating how different types of relationships and parameters affect the algorithm could be the subject of future work.

3.5 Global disambiguation

If a tweet has been identified as ambiguous locally, the algorithm tries to disambiguate it using its global context, which is given by its disambiguated neighbors. Since the global graph is built incrementally, it could be either that when an ambiguous tweet is inserted into the system it is immediately connected to other disambiguated tweets, or that an ambiguous tweet does not have disambiguated neighbors initially and then other disambiguated tweets are connected to it or ambiguous neighbors are disambiguated.

In any case, the global disambiguation phase happens (or re-happens, if the previous attempts have failed) when at least a new disambiguated tweet is connected to an ambiguous tweet.

This phase is crucial and translates one of the most important hypothesis at the base of the algorithm: a tweet will be connected more often to related tweets and will find in them a context which, with highest probability, will allow to choose the correct location among the set of candidates.

**Ambiguous locations** An ambiguous tweet is characterized by a non-empty set $C$ of candidate locations, for which is not possible to find relationships locally.

**Global context of a tweet** The global context of a tweet $t$ is defined taking the set of locations:

$$DN(t) = \bigcup_{t_j \mid (t, t_j) \in E \land t_j \in DIS} loc(t_j)$$

48
and combining them exactly as when creating the local context for a single tweet (Section 3.3). Notice that necessarily $DN(t) \neq \emptyset$.

As in the local case, this will result in a relation $P(t) = \{(l_{1P}, l_{1C}), (l_{2P}, l_{2C}), \ldots\}$ and in a context, that now is \textit{global}:

$$g^*(t) = (DN(t), P(t))$$

Even if in this case there are more different locations than when creating the local context (i.e. $|DN(t)|$ can be greater than $|C(t)|$), there are also two important simplifications:

- each $l \in DN(t)$ is a single and certain location, while each $C \in C(t)$ is a set such that $|C| \geq 1$.
- Many of the relationships are already computed locally. Indeed, the following holds:

$$\forall t_j \mid (t, t_j) \in E, P(t_j) \subseteq P(t)$$

Therefore, only the subset of additional locations must be computed.

In the current implementation the number of distinct neighbors which provide the same location is stored as an attribute node “occurrences number”, for each location, because it is used in the disambiguation phase (see next paragraph).

\textbf{Disambiguation in the global context}  Given $t \in AMB$, the set of candidate locations $C(t)$ and the global context $g^*(t)$, the phase of global disambiguation must find

$$L(t) = \forall C_i \in C \left( \arg \max \limits_{c_j \in C_i} D(c_j, g^*(t)) \right)$$

Where $D(c_j, g^*(t))$ is the disambiguation function that, in general, assigns a score to each candidate location $c$ depending on the global context $g^*(t)$. Therefore, for each set of candidate locations $C$ the candidate location with the highest score is selected.

In general, the disambiguation function could be complex and could take into account different aspects of the context.

In the current implementation, for simplification and performance reasons, the following strategy has been adopted:

- First of all, the inclusion is checked, that is $c \in C$ is selected if $c \in DN(t)$.
  This condition is defined as the maximum value for $D$. If both $D(c_i, g^*(t))$
and $D(c_j, g^*(t))$ hold, with $i \neq j$, the candidate location with the maximum occurrences number is selected.

- As second attempt, is selected $c \in C$ which have a hierarchical relationship with some node in $DN(t)$. This check is the same kind of check described creating a local context (Section 3.3). This condition is defined as an intermediate value for $D$. Again, if more than one candidate locations for a set $C$ satisfy the condition, the candidate location with the maximum occurrences number is selected.

- In any other case $D(c, g^*(t)) = 0$.

Notice that the current implementation allows transforming a general maximum problem in the problem of finding the first positive outcome in a series of ordered attempts, with a major benefit on performance.

### Globally disambiguated node

After the global disambiguation phase

$$L(t) \neq \emptyset \iff \text{the node has been disambiguated}$$

As a consequence, the sets $AMB$ and $DIS$ are updated.

Notice that, exactly as in the local disambiguation case, it could be that even if $L(t) \neq \emptyset$, there are some candidate location sets without any of their candidate locations in $L(t)$. As in the local disambiguation case, the tweet is anyhow considered disambiguated and there are no further attempts for those candidate locations.

### Not globally disambiguatable node

It could happen that $L(t) = \emptyset$ and therefore $t$ remains ambiguous. In this case, it is said that the disambiguation attempt has failed. If so, there will be new attempts once (and if) new tweets will connect to $t$ causing a changing in its context and providing new chances for disambiguation. With some optimizations, the overhead for these cases is greatly reduced. Optimization details are given in 3.7.1.

This case could introduce a problem similar to the problem described in Section 3.3 for local disambiguation, when some candidate location sets are not disambiguated. Indeed, unless a method to decide if a surface form is certainly “not disambiguable” is introduced, the global disambiguation phase will be tried an

---

7 The number of distinct neighbors which provide the same location.
unbounded number of times. This scenario, even if theoretically does not pose any problem, could bring to violate the online constraint. Therefore, it has been addressed in the algorithm allowing an ambiguous tweet, as a whole, to change its state to missing location. This transition is, in general, function of the past disambiguation attempts and their contexts. Introducing this transition has important implications on the performance of the algorithm, which are detailed in Subsection 3.7.1. In the current implementation, a counter parameter DisambiguationCounter has been introduced such that when the number of disambiguation attempts exceed it, the state transition \( \text{AMB} \Rightarrow \text{MIS} \) happens\(^8\).

### Context of a globally disambiguated node
Once a tweet has been disambiguated, it acts exactly as a locally disambiguated tweet to provide a context to its neighbors. The context is given by the disambiguated location \( L(t) \) and by those locations in \( \text{DN}(t) \) which, directly or indirectly include the locations \( L(t) \), that is by each location \( l \in \text{DN}(t) \) such that \( \exists m \in L(t) \mid H(l, m) \) (iteratively updated until the fixed point is reached). This choice avoids the propagation of unrelated locations by the disambiguated nodes since each disambiguated node only contributes with its own locations and the subset of those locations that have been useful for its own disambiguation.

### 3.6 Inference of missing locations

The algorithm is able to address at the same time location disambiguation and location inference. This is typical of algorithms which focus on creating a context, like [22], since once the context is created it can be exploited to infer a missing location besides disambiguating an uncertain one. Handling disambiguation and inference in the same way, inference is a more difficult problem, since the set of candidate locations is empty \( (C(t) = \emptyset) \) and therefore the neighbors does not constitute just an insight but represent the only resource.

The inference phase concerns a missing location tweet, that could be a tweet originally with no candidate locations but also a tweet considered ambiguous at first which has changed its state later after some failed disambiguation attempts.

Since a missing location tweet has no locations and no candidate locations, and potentially the only resource is constitute by its neighbors, the only “ingredient”

---

\(^8\)Since the trigger to a new disambiguation attempt is a changing in the context, all the disambiguation attempts are certainly different one from each other.
3. Geolocation

to inference is the *global context*.

Exactly as for the global disambiguation phase, the inference phase is triggered the first time when the missing location tweet is connected to at least a disambiguated tweet (that could precede or follow it in time), and, if it fails, is re-executed each time the global context changes.

**Inference function** The inference function $I$ is the function that, given the list of disambiguated neighbors returns the actual location of the tweet. Therefore the inference problem can be defined as finding the locations of the tweet $L(t)$:

$$L(t) = I \left( \{ t_j \mid (t, t_j) \in E \} \right)$$

Different inference functions could be applied and have been applied in the literature (e.g. [36]).

The current implementation has not focused on the inference function itself (and in the inference phase in general, which is more a “by-product” of the disambiguation phase) — the function implemented chooses $L$ as the most frequent location among the disambiguated neighbors. Such function is easy to compute and returns always a result. Of course, a more complex function could be implemented without changing the algorithm in its entirety, but it is out of scope and the adaptation of existing inference functions to this algorithm could be a future direction of this work.

3.7 Discussion

After having described the algorithm in the previous sections, some remarks, optimizations, limitations and comparisons are discussed in this section.

First of all algorithmic optimizations are detailed in Subsection 3.7.1. Then, the main limitations (from a theoretical point of view) are detailed in Subsection 3.7.2. The state diagram of a tweet during the execution of the algorithm is summarized in Subsection 3.7.3. In Subsection 3.7.4 the algorithm is compared with other approaches in the literature described in Chapter 2.

3.7.1 Optimizations

The geolocation algorithm has been described, up to now, mainly from a theoretical standpoint, specifying the current implementation when the algorithm
leaves room for different possible solutions.

The algorithm is designed to be online, in the sense of being able to process each new tweet, on average, in a bounded time (that is, independent from the number of tweets already in the system).

Even if some low-level optimizations have been adopted in the system, like the caching of the resources that need an access to disk (like the queries to GeoNames) and the use of data structures and indexes which allow to perform in a constant or linear time the majority of the computation, these optimizations are not the focus of the current work.

Conversely, there are some optimizations at an algorithmic level which must be described because are necessary to satisfy the online requisite mentioned. They are mainly upper bounds to situations where, theoretically, there could be unbounded samples to process.

**Upper bound on the number of neighbors which constitute the context of an ambiguous node** Theoretically, a node could have an unbounded number of disambiguated neighbors (hence graphs) to process. To limit this number, a parameter has been introduced. If the disambiguated neighbors exceed this parameter, they are sampled to match it. The reason behind this choice is the following: if originally the majority of neighbors provide the “right” context, on average the majority of a random subset of them will provide again the “right” context in the majority of cases.

**Upper bound on the number of disambiguation attempts for an ambiguous node** As pointed out in Section 3.5, in the scenario where all candidate locations of a node are false positives or are not available in the external knowledge (GeoNames), the algorithm would repeatedly try to disambiguate the node each time one of its neighbors is disambiguated.

In theory, this could bring to an unbounded number of disambiguation attempts once a node is disambiguated, equal to the number of its ambiguous neighbors, not respecting the online constraint because this number of ambiguous neighbors could increase indefinitely without never reducing.

To overcome this problem, as discussed in Section 3.5, the algorithm allows an ambiguous node to change its state to missing location node.

Therefore, in the current implementation the counter DisambiguationCounter has been introduced so that at most DisambiguationCounter attempts are made on an ambiguous node.
3. Geolocation

It is possible to demonstrate that this optimization allows to perform on average a number of disambiguation attempts which is less than one for each tweet inserted into the system:

Proof. Even if the number of disambiguation attempts once a node is disambiguated is theoretically unbounded, globally at most

$$NA = DisambiguationCounter \times |A|$$

attempts will be tried during the entire execution of the algorithm, where $A$ is the set of tweets initially considered ambiguous. Since $DisambiguationCounter$ is a constant and typically will hold $DisambiguationCounter \ll |A|$, $NA \sim |A|$. Since $A \subseteq V$, the average number of disambiguation attempts per node will be

$$\frac{NA}{|V|} \sim \frac{|A|}{|V|} \leq 1.$$

**Incremental global context creation** As described in Section 3.5, it could happen that an ambiguous node is not disambiguated because the global context does not provide a useful context. This requires to retry the global disambiguation phase, and this is done a constant number of times for each node as seen in the previous paragraph. However, an observation that leads to an optimization is that, given the global context at the $n$-th attempt $g_n^*(t) = (DN_n(t), P_n(t))$, it will always hold

$$DN_n(t) \subset DN_{n+1}(t), P_n(t) \subset P_{n+1}(t)$$

since a new attempt is made when a new disambiguated neighbor is connected and it provides at least a new location, while all the neighbors already connected will be the same of the previous attempts\(^9\). Therefore, in case of a failed attempt is possible to store on the node the computed context $g_n(t)$ so that on the next attempt only the relations between the newly connected node and the stored context must be tested.

---

\(^9\)In the current implementation this holds and it will be assumed in the following.

\(^{10}\)Indeed, there are two exceptions: when a neighbor that has exceed the sliding window has been removed by the global graph $G$ and when the optimization based on $MAX\_NEIGHBORS$ is used and the maximum number of neighbors has been reached, since neighbors start to be sampled and could change from attempt to attempt. However, is easy to check if one of these cases is interested and deactivate the incremental global context creation for those cases.
3.7.2 Limitations

A discussion about the performance of the algorithm in real cases will be done after the evaluation of the algorithm performed in Chapter 5 that will include precise assessments of precision, recall and accuracy in solving geo/geo and geo/non-geo ambiguities.

The aim of this subsection is to outline the main theoretical limitations of the algorithm.

A first limitation is that a location is recognized, and therefore accepted, only if included in the external knowledge (the gazetteer). Therefore, the performance of the algorithm depend on the choice of the gazetteer. The current implementation relies on GeoNames and geolocation inaccuracy have been found in it [78]. However, this is unavoidable since “the output of any geocoding algorithm is only as exact as the knowledge base that underlies it” [78].

A related problem involves the types of locations which can be disambiguated/inferred and their granularity. From a theoretical point of view the algorithm does not pose any restriction in this sense, however a gazetteer such GeoNames for example does not include many names of streets (or does not include hierarchical relationships for them), therefore the current implementation cannot handle street names. A similar limitation holds for building names, etc.

Even if the algorithm is expressly designed to be language-independent, the gazetteer could affect the performance on different languages since its coverage could be uneven.

Besides these kinds of problems related to the external knowledge, the main theoretical limitation of the algorithm is given by its applicability.

Differently from many geolocation algorithms (ref. Section 2.4) this algorithm is not designed to be general purpose. For example, it cannot geocode toponyms in a traditional text or even from a set of randomly sampled tweets, because it would mostly fail in finding contexts. This is because it is expressly designed to be effective in an environment where many small, individually decontextualized but globally and geographically coherent messages are available in a relative short timeframe, that is, for example, in crisis scenarios like natural disasters. This limitation corresponds to the main advantage of the algorithm, which is designed to exploit well known features of social media during such events. Indeed, as shown in Chapter 5 the algorithm will have some behaviors which contradict the traditional toponym resolution task but, in the particular context, represent an “added value”. In particular, some toponyms will not be recognized as locations
3. Geolocation

even if they are actually locations, because they are mentioned “accidentally” and are not part of the event. Since such locations are not interesting to enhance situational awareness about crisis scenarios (and, instead, act more as noise) their elimination can be considered an “added value”. This discussion is detailed in Subsection 5.3.3.

3.7.3 State diagram of a tweet

In this subsection the state diagram of a tweet is depicted in order to analyze and summarize the possible behaviors from the tweet’s state standpoint.

Recall that the possible states are: AMB (ambiguous), MIS (missing location), DIS (disambiguated), INF (inferred).

From the description of the algorithm made in the previous sections the possibility of all and only the following transactions derives:

- **AMB ⇒ DIS**
  - When a tweet is disambiguated in the local context
  - When a tweet is disambiguated in the global context
- **MIS ⇒ INF**
  - When a tweet is inferred (necessarily in the global context)
- **AMB ⇒ MIS**
  - When a tweet is not globally disambiguable.
- **AMB ⇒ AMB**
  - When a global disambiguation attempt fails.

**AMB and MIS are start states.**

Notice that starting in a missing location state, the only possible “positive” outcome is the inferred state, while starting from the ambiguous state there are two possible “positive” outcomes obtainable through the transactions **AMB ⇒ DIS** and **AMB ⇒ MIS ⇒ INF**, thanks to the possibility explained in Section 3.5.

The only self-loop of the diagram, **AMB ⇒ AMB**, is bounded in the maximum number of times it can be followed\(^{11}\).

\(^{11}\)This is due to the “upper bound on the number of disambiguation attempts for an ambiguous node”.

56
3.7. Discussion

All the states are potentially final states. However, excluding nodes without disambiguated neighbors (“isolated nodes”), MIS cannot be a final state. This is summarized in Figure 3.6.

![State diagram of a tweet](image)

Figure 3.6: State diagram of a tweet.

3.7.4 Comparison with other techniques

In this subsection a comparison of the algorithm with other techniques is detailed, focusing in particular on those described in Chapter 2 (state of the art).

The global graph is managed like the graphs described in other systems to detect events through clustering, e.g. [10, 17, 56, 59]: the idea is to connect similar (that is, related) tweets.

However, there are two main differences:

- here the goal is not clustering, but only obtaining a context for each tweet and allowing information propagation;

- the geographical features are never used to connect similar tweets, as for example in [59] because the goal is precisely to find them.

This brings several consequences, mostly related to robustness:

- the local properties of each node gain more importance (each node “sees” only its neighbors).

- The geographical features often represent a bottleneck to perform tweet clustering since they are a rare resource in Twitter (that is, the large majority
of tweets does not come with an attached location [40]. Therefore, instead of using the geographical correlation among tweets to detect events, the idea is to use the other kinds of correlations which are always computable (semantic, social, temporal) to improve the geolocation and only then use the geolocated tweets to perform location-based event detection or other tasks (in Chapter 4 is detailed how geolocation can be inserted into a complete system to improve other tasks).

- Since the neighbors of a node provide its context, what really matters is the majority of neighbors. This means that false positive links (i.e. between not really similar tweets) are absolutely irrelevant if the majority of links are true positive. This is a fundamental difference with respect to using links for clustering: in the latter case each link gives a contribution to a global decision, therefore a set of wrong links could worsen (even slightly) the global clustering result; in this case instead there is a local choose among a set of candidates that can be only right or wrong, and will be right if the majority of neighbors provide the right context, making false positive links irrelevant in those cases.

Therefore, even if the way in which the global graph is managed is similar to works like those, the target is totally different since it shifts from event identification to location disambiguation and inference.

In this sense, the resulting graph has a role which is more similar to the social networks used for location disambiguation and inference in works like [16, 22, 36, 61]. However, they address user geolocation and are based on explicit (articulated) networks, like friendship. Instead, the proposed algorithm addresses message geolocation and the resulting graph has a role of implicit (behavioral) network inferred from communication patterns. Indeed, “implicit networks are particularly interesting in the crisis scenario because many exchanges happen among people who were not connected before the crisis” [13].

Moreover, most of the works based on social networks focus on the inference problem, for example through label propagation [36], while the current work focuses on toponym disambiguation and inference. In this sense, the work is more similar to [22], which addresses toponym disambiguation (and inference), but it focuses on the user profiles rather than individual tweets.

The use of hierarchical relationships among locations is similar to other methods for geolocation both network based [22] and text-based [33].

Many of the cited algorithms for user geolocation based on networks does not
assume to work in an incremental and online environment but to have the social graph available from the beginning. In contrast, the proposed algorithm is designed to work incrementally and online.

To the best of this author’s knowledge no other techniques in the literature have been proposed for incremental and online location disambiguation and inference exploiting behavioral social networks and focusing on contents (i.e. messages) rather than users.
3. Geolocation
Chapter 4

Processing pipelines and evaluation methods

Systems for processing social media contents during natural disasters (and crisis scenarios in general) are usually articulated in several phases \[13, 29\]. Depending on the goal, systems described in the literature have been designed differently, even if there are typically elements in common as pointed out in Chapter 2 (state of the art).

The goal of this chapter is in the first place to present a typical system for social media analysis during crisis scenarios, focusing in particular on the role of the geolocation phase and in its relationships with the other tasks.

Then, several processing pipelines will be proposed. They represent possible processing paths for typical applications in the crisis management area and they all contain a geolocation phase because the goal is to demonstrate its usefulness in this context.

Indeed, in this context, geolocation is typically an enabler or an enhancer for other phases since it can address the scarcity of geotagged messages \[40\] because “while explicit metadata about locations may be absent, many messages in social media do contain implicit references to names of places” \[29\] and can exploit other types of locations not represented by geotags. If on one side geolocation can be useful by itself, for example to populate the maps of a live-monitoring system for social media, on the other side it is typically part of more articulated systems and its effectiveness in crisis scenarios is indirect.

For this reason, after have outlined how geolocation can be exploited for crisis management applications in the current chapter, it will be evaluated in Chapter 5 not only as an isolated task, but also in the context of the proposed geolocation-
4. Processing pipelines and evaluation methods

In Section 4.1 the main tasks and applications of a typical system for social media processing in crisis scenarios are outlined. Then, in Section 4.2 some processing pipelines are proposed to show how geolocation can be useful in different applications. The chapter is concluded in Section 4.3 by a discussion of the evaluation methods of the proposed systems, that will guide their evaluation in the next chapter.

4.1 Phases overview

Figure 4.1 shows the typical phases in a system for social media processing in the context of crisis management, with their mutual relationships.

Notice that typically real systems, like those described in Chapter 2 only focus on some of this phases, especially in terms of those phases which generate an outcome, and that many variations and additions are possible, therefore this diagram does not pretend to be exhaustive but only to sketch typical scenarios.

In particular, it is possible to distinguish:

- a set of preliminary stages, which always include a *data acquisition* phase and often a *data preprocessing* phase;
- a set of processing stages, which may vary and aim to extract and enrich the messages with semantic information, which can be useful by itself, be useful for the data acquisition (*feedback*) or for a series of applications;
- a set of applications which exploit the data to perform something for the end-users, like event detection, summarization, image analysis, etc.

These phases are detailed in the following. Many examples for each phase have been described in Chapter 2, therefore in the following the focus is on their organization and their mutual relationships.

4.1.1 Data acquisition and preprocessing

Data acquisition is performed through the APIs (Application Programming Interface) made available by social media.

The main challenges in this phase are:

- scale, since social media are an example of *big data*;
Figure 4.1: Typical phases in a system for social media processing in the context of crisis management. The relationships model the information flow.
4. Processing pipelines and evaluation methods

- limitations imposed on the amount of messages that can be queried;
- limited expressiveness of the query interfaces.

The same social media can have different kind of APIs to access its data. For example, Twitter has a “search APIs”, through which is possible to search for past messages and a “streaming APIs”, which allows to subscribe to real-time data. In both cases, it is possible to filter the results in terms of keywords, locations and time. Using the “streaming APIs” Twitter allows to retrieve a random sample of 1% of all postings. Differently from Twitter, most large social media platforms does not offer this level of data access publicly [29].

Other challenges related to data acquisition are those related to the data storage itself and its indexing [13].

After messages have been acquired, they are typically preprocessed. This involves steps like:

- **Deduplication**: searching for identical messages or near-duplicates, for example through clustering techniques. In many social media, like Twitter, the “reposting” action by a user has a semantic meaning therefore before removing duplicates their semantic impact should be taken into account for example in terms of social relationships they create, relative importance they give to a message (a message reposted by many users is often considered more relevant), etc., so that there is no information loss for the following phases.

- **Spam removal**: spam and bots detection are well-known challenges in social media. In particular, when a popular event is in progress, bots use hashtags and keywords about the event to increase their visibility and therefore their messages are easily captured.

- **Postfiltering**: this varies depending on the applications and involves text cleaning, further keyword-based filtering, stop-words removal, etc. In some cases more sophisticated semantic-based techniques are employed, like classifier-based filtering, NLP (Natural Language Processing) techniques, etc., however in this description these more sophisticated techniques are considered part of the data processing (see next subsection).

- **Feature extraction**: many processing techniques like clustering, classification etc. operate on vectors, which must be extracted by data through a process of feature extraction. The process of selecting the features to be extracted
is called feature engineering. In the context of social media, typical features include textual features, like $n$-grams\(^1\) and metadata features, like author, timestamp, number of views etc. In Subsection 2.3.1 several examples of feature extraction steps and representations described in the literature are reported.

At the end, the data ready to be analyzed and processed should be available for the following phases.

### 4.1.2 Data processing

Data can be analyzed and processed in several different ways.

A data processing phase aims to extract new knowledge from the preprocessed data, either to present it to the end-users or to assist other processing phases and end applications.

The diagram in Figure 4.1 shows several examples (not exhaustive) of processing steps:

- **Classification** aims to categorize content in a set of classes. Several different classification types have been employed in this context (see Subsection 2.3.1 for examples). Often, classes refer to semantic concepts so that the classification step involves a semantic enrichment. Clustering can be taught as an unsupervised classification, while traditional supervised classification relies on a training set on which the classifier is trained. Once a tweet is classified, the class represent a new feature that can be exploited for different purposes, like event detection \[15, 65\], situational awareness \[77\] or information extraction \[30\]. In several systems classification is used as a first processing step and only messages labeled with certain classes are processed in the following stages, so that the classifier acts as a filter \[13\]. The main advantage of a classifier-based filter over a pure keyword-based filter is the added semantic. Other methods to categorize and add semantics to messages are topic models like LDA (Latent Dirichlet Allocation) \[79\], where one or more topics are assigned to each message based on the included words.

- **Natural Language Processing techniques** include several tasks to process the natural language and enrich it with more knowledge. Traditional techniques include part of speech tagging (annotate each word with its part-of-speech),

\(1\)Set of \(n\) consecutive tokens in the text.
4. Processing pipelines and evaluation methods

dependency parsing (syntax analysis to detect the dependencies), sentiment analysis (to analyze the subjective content of messages), named entity extraction (NER) to find occurrences of named entities like people, locations, companies, etc. in the text. Even if each of these tasks produce a potentially useful output by itself, they are mostly used as enrichment tasks for other steps, for example to improve classification, clustering, summarization or information extraction.

- Information extraction means, in general, to produce structured records starting from unstructured content (like tweets). Methods to do this include trained models and clustering [13].

- Geolocation, that is automatically detect the location of a message (message geocoding), related to its content or the posting location, or of a user (user geocoding). This phase is useful as semantic enrichment step given the scarcity of natively geotagged messages on social media [40]. Techniques used in this phase have been extensively described in Section 2.4 and with the new algorithm presented in Chapter 3.

As pointed out previously, it is possible to have a feedback from the processing stages back to the data acquisition. This is because, after some semantic knowledge has been extracted from the previously acquired messages, it can be useful to further acquire new messages with more precision, for example with more accurate keywords after a topic identification processing step, or with a set of more strict locations after a geolocation processing step. This last examples highlights a case in which the geolocation outcome can enhance a different phase. These techniques are called “adaptive acquisition/filtering”.

The outputs of these processing steps can be used to build several kinds of applications.

4.1.3 Applications

Here with “application” is meant any phase which, through one or more data processing steps, produce an outcome typically useful for end users. As said previously, any processing phase could constitute an application for end users, however there exist typical functional units in real systems, the most common of which are:

- Event detection, that is automatically find occurrences of new events. Section [2.1] details the main techniques and several examples of event detection
systems in social media. Typical data processing steps for event detection include classification [65], clustering [10] and topic modeling [17]. The locations of messages can be used for event detection [41, 59, 69] therefore a geolocation processing phase could be employed effectively.

- **Event summarization**, which aims to automatically produce abstractive or extractive summaries of events starting from their related messages to overcome the overwhelming amount of contents typically posted on social media [53]. Processing techniques include clustering [67] and NLP [52].

- **Dashboards** used to monitor the events in real-time. They are “visual displays that provide a summary of social media during the crisis according to temporal spatial, and thematic aspects” [29]. Therefore they typically combine several processing steps like clustering, information extraction and geolocation to provide time series, maps and other kinds of diagrams.

- **Image analysis**, to obtain relevant and useful images among those embedded in posts to support situational awareness. Indeed, messages do not provide only text information, but can include also images, videos, sounds, etc., both directly and referring to URLs. The extracted images could be used directly or integrated in a system with other sources [20]. To be relevant and useful the images should be semantically enriched and geolocation represents a fundamental step [19].

- **Situational awareness**, that is understanding the “big picture” to gather insights during a natural hazard [72]. This is a broad definition which includes any application supporting users during natural disasters.

### 4.2 Location-based pipelines

In this section several processing pipelines are proposed. They could be seen as standalone systems or part of a more articulated system; anyway they all include a geolocation phase since they aim to demonstrate how the algorithm for geolocation proposed in Chapter 3 can be effectively used in several tasks for crisis management. After their description in this section, they will be evaluated in Chapter 5.
4. Processing pipelines and evaluation methods

4.2.1 Location-based event detection

This pipeline, shown in the diagram in Figure 4.2, aims to perform event detection based on a joint contribution of geolocation and classification.

![Diagram of processing pipeline]

Figure 4.2: Processing pipeline for location-based event detection.

The traditional method to perform event detection based on clustering similar tweets have two drawbacks which are important in the context of detecting natural disasters:

- It does not only detect real events (that is, events happening in the real world) but also “social events”, that is events happening only on the social media itself, because they are constituted as well by bursts of connected and similar tweets and could be related to the same keywords used to search for real events.

- If the assumed definition of event entails a precise location \[3\], the fact that two tweets have the same topic does not say anything about their spatially proximity. Indeed, there could be “worldwide events”, that, even if real, does not should be detected assuming such a definition.

In the literature two approaches described to face these problems are: using a trained classifier to distinguish real-word events (e.g. \[10\]) and by performing event detection driven by locations (e.g. \[69\]).

In the first case the classifier allows to filter only tweets related to target events or directly the target events themselves.

In the second case tweets are clustered (typically through density-based clustering) based on their spatial proximity (that is, using as distance their geographical distance) so that a cluster will be inherently precisely located and will correspond...
4.2. Location-based pipelines

to an event. A frequently used density-based algorithm is DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [18]. DBSCAN takes as input two parameters: \textit{min\_sample} and $\epsilon$. A \textit{core point} is defined as a point with at least \textit{min\_sample} other points (including itself) within a distance $\epsilon$. A \textit{border point} is a point that is within a distance $\epsilon$ from a core point, while all other points are \textit{outliers}. Two core points within a distance $\epsilon$ from each other are put in the same cluster, a border point within a distance $\epsilon$ from a core point is put in the same cluster of the core point, all the other points are considered \textit{noise}. Therefore, an event will correspond to a group of “dense” tweets from a spatial point of view. DBSCAN can find arbitrarily shaped clusters, does not require the number of clusters as parameter and naturally handles noise. Many extensions of DBSCAN exist to work in an incremental way, like IncrementalDBSCAN used in [41].

These two approaches, however, have some limitations. The classifier is characterized by all the challenges described in Subsection 2.3.1 in particular domain adaptation, and moreover cannot individuate spatially near tweets (unless the focus is only on a particular location for which the classifier has been trained, but it is not the case generally). A location-based approach needs geolocated tweets to operate and moreover cannot distinguish real-world events (there could be not real-world events focused on a particular location, which would be detected performing a location-based clustering).

Therefore, the design of the event detection method shown in Figure 4.2 is driven by the identification of complementary strengths in these two approaches and the improvement that a geolocation phase could give in terms of amount of precisely located tweets for a location-based event detection approach. In particular:

- the classifier prevents the location-based event detection from identifying not real-world events, even if precisely located;

- the geolocation phase allows the location-based event detection to work on a large amount of precisely located tweets, even if the classifier has filtered out a part of the initial stream because considered not relevant;

- a detected event will be real (thanks to the classifier) and precisely located (thanks to the location-based method enhanced by the geolocation processing step).

A method for event detection based on this pipeline is evaluated in Subsection 5.5.2.
4. Processing pipelines and evaluation methods

4.2.2 Rapid mapping and situational awareness

The pipeline presented in the diagram in Figure 4.3 aims to show how rapid mapping, a fundamental contributor to situational awareness, is enhanced by a geolocation processing step.

Figure 4.3: Processing pipeline for rapid mapping and situational awareness.

Crisis situations call for timely, reliable information to act in a very uncertain environment and certainly maps represent one of the most useful resources [13].

This pipeline aims to demonstrate that geolocation can bring a positive impact on situational awareness, both directly and improving the generation of maps.

Many social media, like Twitter, have natively geotagged content. This means that it is possible to generate maps for situational awareness even without a preceding geolocation step. Therefore, the challenge is to demonstrate an improvement given by the geolocation phase to situational awareness with respect to the same diagram shown in Figure 4.3 without geolocation.

It is possible to identify several limitations of geotags natively associated to tweets which can be overcome by a geolocation processing step, described in the following.

- **Volume**: most of the messages are not geotagged in social media [13], about 2% of tweets are natively geotagged in Twitter [40]. This could become a limitation for medium-small events. Moreover, this can also negatively affect all the following processing step (information extraction, media extraction, event detection...), particularly if restrictive filters are adopted since they further reduce the number of available tweets.

- **Velocity**: independently from the volume, geotagged messages have a velocity limitation. In the case of Twitter, a tweet needs to be posted from the target location and from a GPS-enabled device\(^2\) to be labeled with that location. This means that independently from the number of users and total tweets about an event, the subset of correctly geotagged tweets will be always

\(^2\)Another prerequisite is the explicit consent of the user in sharing his location.
bounded by the number of users physically in the target location with a GPS-enabled device. Moreover, the distribution of users equipped with GPS-enabled devices is not uniform and this could bring to many geotagged tweets coming from places different from the target location simply because there are more users with GPS-enabled devices there. An example of this problem from a real dataset is reported in Subsection 5.5.1. Velocity limitation and skewed distribution of geotagged tweets have also to do with the type of location entailed by geotags.

- **Type**: among the various types of location related to social media [1], the location expressed by a geotag is naturally highly correlated to the posting location, that is the location where the tweet is posted by the author. Even if this type of location is certainly useful for situational awareness in crisis scenarios, another type of useful location is the location related to the message content, and it is neglected by geotags. This kind of location is potentially helpful because it accounts for those tweets which refer to a location without the user being physically in that location with a GPS-enabled device, and allows to address the problems of limited velocity and skewed distribution, as will be shown in Subsection 5.5.1.

A geolocation processing step should be able to overcome these limitations geolocating more tweets with respect to the natively geotagged ones and exploiting also the locations related to the message content near to the posting locations.

Moreover, a geolocation processing step tailored to a specific application context can affect the detected locations privileging those useful for the application context.

The geolocation algorithm proposed in Chapter 3 will be tested against geotags in Subsection 5.5.1 to evaluate this kind of pipeline.

### 4.2.3 Image analysis

The pipeline in Figure 4.4 shows a typical image analysis methodology for social media [19]: the extracted images are geolocalized to be analyzed.

All the limitations of geotags discussed in the context of rapid mapping and situational awareness affect not only information extraction, but also all the attached media like images.

Images are a useful resource that is possible to extract from social media since they can aid the relief effort and can be obtained quicker after the event with
4. Processing pipelines and evaluation methods

Figure 4.4: Processing pipeline for image analysis.

respect for example to images coming from satellites [19].

However, to be relevant, for example to be mapped with respect to the affected areas, images should be precisely located.

Therefore the image analysis, as shown in Figure 4.4 can be enhanced by a geolocation step made at the message level (that is, text-based). Images lose all metadata — including their geographical coordinates — when stored by Twitter, therefore the only alternative would be an image-based geolocation.

In Subsection 5.5.3 the limitations of using only geotags for image analysis will be quantified and will be shown how a text-based geolocation step (that is, the proposed algorithm) is able to overcome them in a case study.

4.3 Evaluation methods

In this section the evaluation methods for the geolocation algorithm presented in Chapter 3 and the pipelines presented in the previous sections are detailed. The evaluation results are in the next chapter.

**Geolocation** Text-based geolocation can be splitted into toponym *identification* and toponym *disambiguation*. The first one refers to finding occurrences of locations names in the text, the second one in disambiguating a toponym individuating the exact location it refers to (and, therefore, the associated coordinates).

Traditionally toponym recognition is evaluated through *precision* and *recall*
4.3. Evaluation methods

Precision is defined as:

$$\text{precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

while recall is defined as:

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

In this context, a true positive is a toponym in the text which is correctly detected to be later disambiguated, a false positive is a n-gram in the text which is wrongly detected as location (this is often caused by geo/non-geo ambiguities, i.e. common words named as locations), a false negative is a toponym in the text which is not detected.

Typically precision and recall are summarized by the F1 score (or simply F-score) which is defined as their harmonic mean multiplied by 2 to scale the score to 1 when both precision and recall are 1:

$$F\text{-}\text{score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Toponym disambiguation is typically evaluated through accuracy, that is the fraction of correctly disambiguated locations.

Evaluating disambiguation, there could be different choices about from which toponyms to start. One could choose the toponyms detected in a previous toponym recognition task or could start from all the toponyms given as input.

In the case of the algorithm proposed in Chapter 3, since it identifies and disambiguates the toponyms at the same time, it has been chosen to compute the accuracy considering only the toponyms identified by the algorithm itself. This means that if a toponym has not been identified, it will not be considered as an error in the disambiguation phase. An error will be a correctly identified toponym which is incorrectly disambiguated, but also a toponym incorrectly identified could bring to a disambiguation error (the only correct disambiguation in this case would be disambiguate to none). Doing this, the disambiguation phase is evaluated only when it can have an impact. Of course, this choice implies that to totally assess the performance of the algorithm both precision/recall for the identification task and accuracy for the disambiguation task must be considered together.

Regarding location inference, the evaluation becomes more difficult because is not always possible to state the true location a tweet refers to when it does not
explicit mention it. For this reason, and also because the current implementation of the algorithm proposed in Chapter 3 has focused mainly on identification/disambiguation, location inference is not evaluated in this thesis and could be subject of future work (ref. Chapter 6).

**Event detection** Event detection is typically evaluated through precision/recall or accuracy. Nonetheless, the fact that it is difficult to precisely annotate events because it is difficult to precisely define them often calls for qualitative evaluations.[8]

In this work it has been chosen to *quantitative* assess the event detection capabilities of the pipeline with different parameters and then *qualitative* discuss how this would reflects in terms of detected events, depending on the definition of event considered. Particular attention will be given to the capabilities of detecting *multiple events* happening at the same time and distinguishing *real* emerging events.

**Situational awareness** It is difficult to formulate a quantitative evaluation for the contributions to situational awareness. However, it is possible “to identify and measure features that could support technology in analyzing mass emergency situations”[72].

In this work it has been chosen to evaluate the contributions of the geolocation phase to situational awareness through some *case studies* on datasets related to real natural disasters. The evaluation will be based on quantitative results and comparisons.

**Image analysis** In the context of image analysis the focus will be on quantifying the amount of images geolocated by the algorithm through the geolocation of the related tweets and compare this value to the number of images natively geotagged.

Since on social media many images are duplicates or are not really useful for emergency management tasks, the analysis will be performed also on a subset of *unique* and *useful* images.
Chapter 5

Experimental evaluation

In this chapter the evaluation results are detailed and discussed.

First of all, in Section 5.1 the datasets used in the rest of the chapter are presented. Then the algorithm for geolocation presented in Chapter 3 is evaluated in Section 5.3 and Section 5.4. Given the importance of geolocation for other applications performing crisis management, the algorithm is evaluated in the context of the processing pipelines proposed in Chapter 4 in Section 5.5. Each section includes a discussion about the results.

The metrics and the methods used for evaluation have been detailed in the previous chapter (Section 4.3).

5.1 Dataset

Two datasets are considered in the following: AMATRICE and FLOODS.

The AMATRICE dataset stores the tweets posted immediately after the earthquake of the 24 August 2016 in Central Italy. It contains about 48 hours of tweets.

The FLOODS dataset stores tweets about floods, hailstorms, hurricanes and landslides happened in Italy from 14 October 2016 to 16 November 2016. All the days have been acquired entirely. Several small-medium scale events occur in this dataset.

Tweets are retrieved simply searching for keywords related to the target events.

Both datasets contain only tweets in Italian (according to the filter implemented by the Twitter crawler).
5. Experimental evaluation

5.2 Parameter setting

With respect to the parameters introduced in Chapter 3, the following choices have been made: the threshold for the similarity among tf-idf vectors has been fixed to 0.4, the maximum number of disambiguated neighbors considered and the maximum number of disambiguating attempts have been fixed to 10. These values have been obtained empirically during a training phase, similarly to [10, 56]. Moreover, the tf-idf vectors have been compared without being previously hashed (like for example in [56]) to assess the performance of the algorithm itself without introducing approximations in the comparisons.

As explained after the description of the algorithm, many variations in terms of relationship types and parameters are possible, and approximation techniques for the comparisons could be employed. Their analysis could be the subject of future research.

5.3 Toponym recognition

The algorithm proposed in Chapter 3 addresses location disambiguation (that is, linking a mention to the exact location it refers to in the gazetteer), assuming that the set of candidate toponyms is given — that is, the named entity recognition (NER) task is performed previously. However, for its fundamental characteristic of privileging intrinsically the precision over the recall, the algorithm could theoretically accept as input — maintaining good results — simply all the possible n-grams, that is working with a “dummy” NER module which achieves the maximum recall simply marking each n-gram as candidate location. The limitations of this approach would be mainly on the performance side, since one of the assumptions is to have a “reasonable” number of candidate locations. In such a scenario, the algorithm would actually perform toponym recognition and location disambiguation as a joint task. In practice, as explained, it cannot be considered a toponym recognition algorithm by itself, but it acts as a “refiner” in the toponym recognition phase because it does not take blindly as correct all the toponyms given as input, but only those for which it finds a context (disambiguating them.

---

1 Toponym recognition task in this case, since only locations are considered as named entities.
2 A location is disambiguated if and only if a context is found.
3 The number of candidate locations in certainly bounded since the tweet’s maximum length is fixed, however, as discussed in Section 3.3, a NER library will provide a number of candidates which is much less than the number of all possible n-grams, and this affects the performance.
at the same time). The algorithm will take as input the toponyms given by a NER library, and, as explained in Section 3.3, a high-recall setting for the library should be chosen given the “refiners capabilities” of the algorithm. In any case, is interesting to measure the performance obtained by the algorithm in the toponym recognition task only, without considering location disambiguation (which is evaluated in the next section).

In the following, first of all a ground truth for the evaluation is built (Subsection 5.3.1). Then, as baselines, two state-of-the-art NER libraries are evaluated with different parameters (Subsection 5.3.2). The algorithm is evaluated in Subsection 5.3.3. A discussion about the usefulness of the locations recognized by the algorithm and the related implications is in Subsection 5.3.4.

5.3.1 Ground truth

On the AMATRICE dataset (limited to the first 24 hours), a random sample of 300 tweets have been selected, which constitute the SAMPLE dataset. Each tweet in the SAMPLE dataset have been manually annotated with the toponyms mentioned in the text. This constitutes the ground truth for the following evaluation. The manual annotation allows to precisely evaluates the results obtained by the algorithm and by other two baselines. Notice that, even if the algorithm is evaluated on the SAMPLE dataset, it has been executed on the entire AMATRICE dataset because for its correct execution all the tweets in temporal order are necessary. Each tweet have been tokenized and all the toponyms have been annotated, except states (e.g. Italy), which are considered too coarse-grained. The tokens constituted only by digits and punctuation marks have been removed. At the end, 189 toponyms have been marked among 3812 tokens on the 300 tweets.

5.3.2 Baselines

First of all (Test-Default), two state-of-the-art NER libraries [2,6] have been employed to find all the toponyms mentioned in the tweets. They are general purpose libraries which contain pre-trained models on the Italian language (since the tweets in the datasets are in Italian). Both the libraries have been used simultaneously to maximize the coverage (that is, a candidate location is considered when returned by at least one of the two libraries). In this first experiment both the libraries are configured with their standard parameters which should correspond to the best trade-off between precision and recall.
5. Experimental evaluation

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test-Default</td>
<td>0.83</td>
<td>0.67</td>
<td>0.74</td>
</tr>
<tr>
<td>Test-Recall</td>
<td>0.40</td>
<td>0.93</td>
<td>0.56</td>
</tr>
<tr>
<td>Test-Algorithm</td>
<td>0.99</td>
<td>0.52</td>
<td>0.68</td>
</tr>
</tbody>
</table>

Table 5.1: Precision, recall and F-score in the toponym recognition task for Test-Default, Test-Recall and Test-Algorithm.

The results are shown in the first line of Table 5.1. The overall F-score is 0.739 and it is possible to notice that the result is “recall-bounded”. This is in line with the results obtained in the literature [60]. One of the reasons for the low recall is the noisy nature of tweets (as discussed in Section 2.4), in particular the informal language, misspells and bad capitalization. Moreover, in several cases locations are inserted as hashtags, which are typically disconnected from the rest of the sentence.

For example, in the following tweet the location “Porto Potenza” is not recognized (notice the bad capitalization):

botta di terremoto bella forte porto potenza mc

and similarly in the following tweet:

#terremoto magnitudo 6.1 vicino #terni. Scossa percepita anche nelle #marche e nel #lazio!

As second experiment (Test-Recall), one of the two NER libraries mentioned above [2], which allows an “extra-recall” parameter, has been configured to significantly privilege recall over precision

(while the other library is used again with its default configuration, and both have been used also in this case). This has been done to measure the impact on the precision of such a choice in this context, and also because, as said previously, a high-recall set of candidate locations is preferable as input for the geolocation algorithm.

The results are shown in the second line of Table 5.1. Even if the recall increased significantly, the precision decreased more in proportion with respect to the previous default scenario, bringing an overall lower F-score of 0.56. This second result is overall worse than Test-Default, but given its high recall constitutes a good input for the algorithm.

---

4 This negatively affects NER libraries since they are based on models which consider the syntactic context of a word to assign it a label.

5 “Extra-recall” = 0.75.
5.3.3 Proposed algorithm

The geolocation algorithm has been tested using the candidate locations obtained in Test-Recall as input. Without considering the disambiguated locations themselves, but only the surface forms which link to any location, that is only the “toponym recognition part” of the algorithm, the results shown in the third line of Table 5.1 have been obtained (Test-Algorithm).

The following considerations arise from the results:

• First of all, it is possible to notice that even starting from a set of candidate locations with a low precision (≈ 0.4) the set of locations matched has a precision of 0.99. This means that the algorithm is able to solve almost-completely geo/non-geo ambiguities.

• The trade-off is a lower recall — which, however, remains > 0.5, that brings the overall F-score to 0.685. The loss in terms of recall impacts less than the gain in terms of precision, since the overall F-score outperforms the F-score of the starting point Test-Recall.

• Even if the F-score of Test-Algorithm is lower than Test-Default, the algorithm has also disambiguated the toponyms at this point (the disambiguation is evaluated in the next section) and it should be taken into account to assess the overall performance of the algorithm; instead Test-Default refers to the toponym recognition task only, with the disambiguation task still to do, therefore the results are not directly comparable. Moreover, a discussion about the usefulness of the identified locations must be considered to understand one of the reasons of the false-negatives of the algorithm (and therefore its lower recall) — see Subsection 5.3.4.

5.3.4 Usefulness of the identified locations

Usually, a toponym recognition task (and, in general, a Named Entity Recognition task) does not take into account the effectiveness, in terms of “usefulness”, of the identified locations — that is, any toponym in the text should be matched.

However, in a context like the applications described in Chapter 4, where the focus is on the events and their description, there is a reason to question if all the locations are equally important and useful, and, moreover, if some locations have a role which is closer to noise than information. To show this, consider the following tweet extracted from the SAMPLE dataset:
5. Experimental evaluation

Terremoto nel centro Italia. Agrigento pronta ad aiutare le popolazioni colpite

This tweet refers to a location which is clearly not affected by the event, even if the tweet is related to the event in general (to offer help). This means that this tweet should not contribute with its associated information, media and meta-data to the description of the locations affected by the event, and, instead, its presence could conversely act as noise for the event.

In a traditional setting, “Agrigento” should certainly be matched as location by a NER library (and, indeed, it is matched both in Test-Default and Test-Recall) and then linked during the location disambiguation phase, however as explained is questionable if its detection and disambiguation is useful in a system like the one described in Chapter 4.

This highlights one of the key points of the geolocation algorithm described in Chapter 3: it is not a general purpose algorithm, and this specificity allows it to perform better (or exclusively) in a particular domain, which is during a precisely-located event. Indeed, the proposed algorithm does not recognize “Agrigento” in the previous tweet and this happens by-design, since it is impossible to find a context for this location, being it outside the event.

Therefore, one of the reasons of the lower recall of the algorithm (Table 5.1) is certainly the set of false negatives corresponding to real — but not useful — locations.

To quantify this, the SAMPLE dataset has been re-annotated to keep only the useful locations, that is the locations related to the earthquake (not necessarily damaged locations but where the earthquake was actually felt). Doing this, 26 toponyms have been removed by the ground truth because not useful, and the toponyms in the dataset passed from 189 to 163. This constitutes the ONLYUSEFUL dataset.

The tests Test-DefaultUseful, Test-RecallUseful and Test-AlgorithmUseful have been performed on the ONLYUSEFUL dataset, following the same procedures of Test-Default, Test-Recall and Test-Algorithm respectively. The results are shown in Table 5.2.

The results with and without the not-useful locations are compared in Table 5.3. From this comparison it is possible to notice:

- First of all, Test-Default and Test-Recall are significantly worsened, while, conversely, Test-Algorithm is slightly improved. This can be explained by the fact that a traditional NER algorithm (like those used in
5.4 Location disambiguation

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>R</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test-DefaultUseful</td>
<td>0.71</td>
<td>0.66</td>
<td>0.68</td>
</tr>
<tr>
<td>Test-RecallUseful</td>
<td>0.35</td>
<td>0.94</td>
<td>0.51</td>
</tr>
<tr>
<td>Test-AlgorithmUseful</td>
<td>0.91</td>
<td>0.56</td>
<td>0.69</td>
</tr>
</tbody>
</table>

Table 5.2: Precision, recall and F-score in the toponym recognition task for Test-DefaultUseful, Test-RecallUseful and Test-AlgorithmUseful.

<table>
<thead>
<tr>
<th></th>
<th>All locations</th>
<th>Only useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test-Default/DefaultUseful</td>
<td>0.74</td>
<td>0.68</td>
</tr>
<tr>
<td>Test-Recall/RecallUseful</td>
<td>0.56</td>
<td>0.51</td>
</tr>
<tr>
<td>Test-Algorithh/AlgorithmUseful</td>
<td>0.68</td>
<td>0.69</td>
</tr>
</tbody>
</table>

Table 5.3: Comparison between the F-scores targeting all the locations or only the useful ones during the three tests.

The focus of the algorithm proposed in Chapter 3 is on location disambiguation. To each toponym marked in the SAMPLE dataset the correct interpretation has been assigned in terms of the exact entry in the gazetteer as ground truth. Notice that exist a few locations which are not available in the GeoNames gazetteer and therefore are ignored in the evaluation.

- The improvement in the F1 score of Test-AlgorithmUseful with respect to Test-Algorithm is given by an increase in recall. This “latent recall” obtained can be seen, as explained previously, as an added value of the algorithm in being able to discard noisy location analyzing an event.
- Considering only useful locations, the algorithm has the highest F-score.

5.4 Location disambiguation

The focus of the algorithm proposed in Chapter 3 is on location disambiguation. To each toponym marked in the SAMPLE dataset the correct interpretation has been assigned in terms of the exact entry in the gazetteer as ground truth. Notice that exist a few locations which are not available in the GeoNames gazetteer and therefore are ignored in the evaluation.
5. Experimental evaluation

<table>
<thead>
<tr>
<th>Test-Algorithm</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.97</td>
</tr>
<tr>
<td>Test-MostPopulated</td>
<td>0.83</td>
</tr>
</tbody>
</table>

Table 5.4: Accuracy in the disambiguation task for Test-Algorithm and Test-MostPopulated.

Evaluating the location disambiguation phase there is necessarily an influence of the toponym recognition phase, which constitutes the input. This means that an incorrectly identified location (false positive) could bring to an error in the location disambiguation phase and in this case the disambiguation is considered correct only if it links to nothing. On the other side, a location which is not identified in the toponym recognition phase (false negative) is totally transparent for the disambiguation phase.

For these reasons, it has been chosen in the following to evaluate the disambiguation based on the results obtained in the identification phase. More details about the evaluation methods are available in Section 4.3. However, the classes of errors in the disambiguation phase which are inherited by the identification phase and their impact will be discussed in Subsection 5.4.3.

5.4.1 Baseline

As baseline, a disambiguation based on the most populated entry among the ones matching the toponym has been tested (Test-MostPopulated). Such heuristic models the intuitive fact that a more populated location has more chances to be mentioned, and, in this sense, is better than an approach which simply links to a random location among the ones matching the toponym.

The set of toponyms used corresponds to those obtained in Test-Defaul (see Subsection 5.3.2), since it is the set with the highest F-score.

The Table 5.4 shows the results obtained (line 2). The errors have different sources, which are discussed in Subsection 5.4.3.

---

6A false negative in the toponym recognition phase is certainly an error for the system as a whole, however it is an “unrecoverable error” in the disambiguation phase, in a traditional pipeline composed by identification-disambiguation.
5.4.2 Proposed algorithm

To evaluate the disambiguation capabilities of the algorithm the same test discussed in Subsection 5.3.3 has been considered, without disregarding the disambiguated locations themselves this time.

The Table 5.4 shows the results obtained (line 1). It is possible to notice that the high precision in identifying the locations (Table 5.1) goes with a high accuracy in disambiguating them. These two results come from the same fundamental feature of the algorithm which identifies (disambiguating it at the same time) a location only when there is a “positive proof” in terms of context.

5.4.3 Discussion

In the following the results are discussed, focusing on the main classes of errors and providing several examples.

5.4.3.1 Geo/non-geo ambiguities

Toponym recognition plays a key role in solving geo/non-geo ambiguities, since such ambiguities can be solved in the first place avoiding their identification, that is obtaining a high precision in the toponym recognition task.

The precision $\approx 0.83$ obtained in TEST-DEFAULT is indicative of some potential geo/non-geo ambiguities. Some examples are:

Amatrice spezzata dal terremoto: il dramma nel giorno della festa più importante La Repubblica

Where “Repubblica” is marked as location. In this case it is the name of a newspaper, but many places in Italy have “Repubblica” in their names.

mia cugina e la madre stavano in vacanza vicino Amatrice. Quinto piano in hotel e hanno sentito tutto. #terremoto

Where “Quinto” is marked as location. In this case it is simply a common word (accidentally capitalized), but many places in Italy have “Quinto” in their names.

In TEST-RECALL the precision is only $\approx 0.40$ and many potential geo/non-geo ambiguities arise from this. Besides the examples reported above, many common words are tagged as locations.

Notice, however, that some false positives could not bring to geo/non-geo ambiguities and this happens when the false positive cannot be linked to any entry in the gazetteer. An example of this obtained in the TEST-DEFAULT is:
5. Experimental evaluation

Sisma, partite unità Protezione civile Comune e Metrocittà: Sono a
dirette a Rieti come supporto...

where “Metrocittà” is marked as location. No location in Italy has “Metrocittà” in
its name (according to the gazetteer).

Regarding TEST-ALGORITHM, there is just 1 false positive which is a token
“it”, belonging to an URL. The reason is that it matches “Italy” in the gazetteer
and it finds a context (even if then locations at the country-level, like “Italy” are
discarded because too coarse-grained).

5.4.3.2 Geo/geo ambiguities

The disambiguation phase is the only phase which addresses geo/geo ambigu-
ities, on the hypothesis that the ambiguous locations have been identified in the
identification phase.

Errors in this phase are particularly important from a practical point of view in
a system which analyzes mass emergencies in real-time because they correspond to
potentially useful tweets — with their attached information and media — where
the authors have reported the locations, but failing their disambiguation vanishes
the usefulness of the attached information and media. This is because in such a
system typically media and information are useful only when correctly localized
[41], therefore mistakes solving geo/geo ambiguities correspond to “lost chances”.

It is possible to see several examples of errors in geo/geo ambiguities in TEST-
MOSTPOPULATED, which are instead correct in TEST-ALGORITHM, all in the
SAMPLE dataset:

Questa la Salaria SS4 all’altezza di Sigillo ... #terremoto #amatrice
#Accumoli #Posta #ArquatadelTronto

This tweet is particularly interesting because there is an attached image showing
the damages affecting an important road (SS4) (Figure 5.1), and during natural dis-
asters infrastructural damages are one of the most useful type of information [29].
This road is very long (over 200 kilometers) therefore being able to exactly localize
the damages is not trivial. Fortunately, the tweet specifies “Sigillo” as interested
location. However, there are 3 places with this name in Italy and the most popu-
lated “Sigillo” in Italy — municipality in the province of Perugia, population 2468
— is the wrong one, since from the context of the tweet is evident that the correct
one is part of the municipality of Posta, province of Rieti, population 151. Thanks
5.4. Location disambiguation

Figure 5.1: Image attached to a tweet: damages to a road.

Figure 5.2: Local context built from the tweet. It includes all the locations in relationship with other locations in the tweet itself. “Rieti” is not explicitly mentioned in the tweet but it comes from a “two locations with the same parent” relationship.
5. Experimental evaluation

to the local context, the proposed algorithm is able to disambiguate it correctly. The local graph built by the algorithm is shown in Figure 5.2.

Terremoto, le vittime salgono a più di cento. Nuova scossa con epicentro Castelluccio...

This tweet specifies “Castelluccio” as interested location. There are many locations in Italy containing “Castelluccio” and the most populated — province of Potenza, population 2179, is the wrong one. From the context, it is evident that the intended location is Castelluccio part of the municipality of Norcia (population 150). However, in this case a local context is totally missing (Castelluccio is the only location specified in the tweet); the algorithm has been able to disambiguate it using the global context.

Many tweets specify “Arquata”, as location, like:

Terremoto, 73 i morti ad Amatrice, Arquata e Pescara del Tronto

The most populated “Arquata” in Italy is Arquata Scrivia (population 4992), but it is evident that these tweets refer to Arquata del Tronto (population 1287).

Another example is the following:

presunta vittima a #Saletta estratta viva dalle macerie poco fa! Ci sono ancora persone da tirare fuori, avanti così! #terremoto

There are several locations in Italy named “Saletta” and the most populated has 1469 inhabitants (Emilia-Romagna). In this case the local context is missing, but it is evident that the intended location is Saletta part of Amatrice (population 33). Indeed, the global context built by the disambiguated neighbors of the tweet is shown in Figure 5.3, together with the relationship “Saletta” finds in it.

In the AMatrice dataset there are over 20 tweets reporting “Saletta” as location since it is one of the most affected places, even if it has only 33 inhabitants.

These examples show also a typical characteristic of natural disasters: often very small locations becomes suddenly notorious (for example because correspond to the epicenter of an earthquake) and are mentioned many times in a short time-frame. This brings several considerations. First of all, it is not possible to use only a subset of the gazetteer (e.g. “keep only the locations with a population > 500”) making assumptions on the type and dimension of the locations mentioned during an emergency. Secondly, it is not safe to solve geo/geo ambiguities giving much weight to attributes like population or notoriety of a location. Thirdly, it could
5.4. Location disambiguation

Figure 5.3: Global context built from the tweet’s neighbors. “Saletta”, in bold, is disambiguated finding a relationship in this context.

happen that correctly disambiguating a small location becomes crucial to locate and analyze precisely and quickly an event.

All these considerations show that there exist an impact on the usefulness of a disambiguation which varies from case to case. In Subsection 5.3.4 it was pointed out that not all the identified locations are equally useful, since some locations could be strictly related to the event while others could act as noise. Similarly, in a system which detects and analyzes natural disasters, some disambiguations could have a key role and this also depends on the natural disaster itself: for example, if the epicenter of an earthquake happens to be in a small location named exactly as other more populated locations, such disambiguations will be crucial to timely identify correctly the disaster and extract useful insights.

5.4.3.3 Algorithm limits

Besides the limits of the algorithm itself, the current implementation and the gazetteer of choice (GeoNames), discussed in Section 3.7, the evaluation has highlighted an important limit in the performance given by a bounded recall in identifying the locations.

This is due to the fact that the algorithm searches for a “positive proof”, that is a context, to disambiguate (and at the same time to identify) a location and otherwise it simply waits until the context is found. Since the sliding window is limited and a context for a tweet could never be found in that timeframe, many
tweets will be still waiting for a context when they exit the sliding window and at that point their locations will never be recognized giving an amount of false negatives responsible for the lower recall.

In any case, as noticed previously, the recall is still acceptable, specially if only the useful locations are considered and taking into account the fact that this approach brings to a very high precision.

Of course, changing the global graph metrics in terms of types of relationship and thresholds could improve this result, since it has a big impact on the outcome of the algorithm (ref. Section 3.4). This could be subject of future development.

The current implementation has not focused on performance, and, even if the algorithm is designed to be online, a real-time implementation would require specific optimizations. In particular, the similarity between tweets should be computed through high-optimized approximations, like hashing (see Section 3.4) and the database of the gazetteer (GeoNames) should be organized specifically for the algorithm (while, in the current implementation, the default organization is used). Moreover, since many locations are mentioned many times in different tweets and several operations are repeated with the same inputs continuously, the algorithm could benefit from a machine with enough RAM to cache all the results and set up in-memory databases. Moreover, the algorithm could benefit from parallel execution optimizations. However, as reference, the current implementation executed on a laptop with 8 GB of RAM has achieved 4 tweets/second on the AMATRICE dataset (and this includes also the time required to compute TEST-RECALL given as input).

5.4.4 Statistics

Even if the evaluation has focused only on the manually annotated SAMPLE dataset, some statistics about the algorithm outcome on the entire AMATRICE dataset are reported in the following.

Overall, on the AMATRICE dataset (which comprises 130836 tweets), 42231 (32.3%) tweets are disambiguated (that is, at least one location has been disambiguated in the text), among which 36110 (27.6%) using the global context and the remaining only the local context.

However, 42231 tweets are those with locations at any granularity, which includes also state or region. Excluding state and region granularity (often considered too coarse-grained), 26979 (20.6%) tweets are disambiguated (both using the local context and the global context).
Each tweet can be associated with more than one location. The 26979 “fine-grained” disambiguated tweets bring totally 31063 locations\(^7\), that is, on average, 1.15 locations for each tweet.

Among the “fine-grained” identified locations, 286 \((\approx 0.9\%)\) have less than 100 inhabitants, 862 \((\approx 2.7\%)\) less than 500 inhabitants and 18202 \((\approx 58.6\%)\) less than 5000 inhabitants.

As comparison, on the AMATRICE dataset only 0.27\% of tweets are natively geotagged.

## 5.5 Geolocation-based pipelines

After having evaluated the performance of the geolocation algorithm itself, in this section the geolocation algorithm will be evaluated in the context of more complex processing pipelines, as those described in Section 4.2. This will be carried out through case studies focusing at the geolocation step as an enhancer and an enabler for different applications. Comparisons with natively embedded geotags will be made.

### 5.5.1 Geotagged tweets vs text locations

It is useful to compare the locations extracted by the algorithm to the geotags automatically assigned by Twitter in terms of added situational awareness. As explained in Subsection 4.2.2, indeed, the geolocation processing step should be able to overcome the limitations of geotags.

As a case study, the following scenario is evaluated: considering the AMATRICE dataset, the epicenter of the earthquake happened in Accumoli (RI)\(^8\) (and consequently that zone is the most affected), however in the first hours many reports come from the zone of Rome, simply because it is relatively near but much more populated. However, in Rome there have been no major damage. A system which performs event detection and analysis to help the relief effort is typically interested in the most affected areas (like Accumoli in the example) while is only marginally interested in a zone like Rome.

\(^7\)Notice that the algorithm consider as two separate locations two locations which are not in a father-child relationship. For example, if a tweet mention location A and location B, and A is part of B, only A is considered as a location (even if B is recognized), because it is more fine-grained. Instead, if A and B are not in a geographical relationship, both are considered locations of the tweet. The details about this approach are in Section 3.3.

\(^8\)https://it.wikipedia.org/wiki/Terremoto_del_Centro_Italia_del_2016
5. Experimental evaluation

However, the cumulative number of geotagged tweets after the earthquake near Rome and Accumoli\textsuperscript{9} plotted in Figure 5.4 show that the total number of reports coming from Rome is greater than the total number of reports coming from Accumoli for about 5 hours after the earthquake. Later, the number of reports from Accumoli becomes greater, but even after 12 hours the difference is marginal. It is also possible to notice that the total number of reports is small (about 60), because only few tweets are natively geotagged.

![Figure 5.4: Cumulative number of tweets near Rome and near Accumoli according to the embedded geotags.](image)

The graph plotted in Figure 5.4 is hardly helpful to correctly detect the epicenter of the earthquake precisely and quickly, unless it is further processed (for example to weight less those tweets coming from more populated areas). Moreover, since the total number of geotagged tweets is small, it is prone to change substantially based on few tweets and to be affected by noise.

Performing the same experiment based on the text-based locations extracted by the algorithm\textsuperscript{10} the result is shown in Figure 5.5.

The following considerations arise:

- The total number of tweets is significantly (orders of magnitude) more. One of the reasons is the fact that about 1\% of tweets is natively geotagged,

\textsuperscript{9}Here and in the following, a location is considered “near” to a city if it has a distance which is $< 50$ km from the city’s coordinates according to GeoNames. This distance is arbitrary but useful to make qualitative assessments considering the orders of magnitude of the results.

\textsuperscript{10}State and region level locations have been discarded because often considered too coarse-grained.
5.5. Geolocation-based pipelines

but more than 20% of tweets have a location extracted from the text (ref. Subsection 5.4.4).

• After about 3 hours, the total number of reports referring to Accumoli becomes greater than the total number of reports referring to Rome, and this difference becomes more and more greater in the following. Indeed, a fundamental difference is given by the rate of growth.

It is evident that the graph plotted in Figure 5.5 is helpful in distinguishing Accumoli from Rome even without further processing — with a positive impact on situational awareness — and it is more robust to noise.

Comparing the two graphs of Figure 5.4 and Figure 5.5 it is possible to see that a text-based geolocation processing step can overcome the limitations of geotags described in Subsection 4.2.2. In terms of volume, the total number of tweets referred to Accumoli is passed from tents to thousands. In terms of velocity the rate of growth of tweets referred to Accumoli is much more higher with respect to tweets referred to a non-affected location like Rome, and this is certainly also due to the type of location extracted by the geolocation algorithm, which is related also to the message content and not only to the posting location. Exploiting the locations related to the content (even without knowing exactly which they are) allows to naturally select the most discussed locations, which should coincide with the most affected ones during a natural disaster.
Of course, a real system for assessing situational awareness could benefit from using both geotags and extracted locations.

5.5.2 Event detection pipeline

As second case study the FLOODS dataset is considered, focusing on how text-extracted locations could help in location-based event detection.

A deduplication processing step has removed duplicate tweets, considering as duplicates also the retweets (but not the first retweet if the original tweet is not in the dataset).

The graph of the tweets over time (\#tweets per hour) is shown in Figure 5.6. It is possible to notice some bigger peaks, even if, overall, there is a peak corresponding to each day.

As explained in the pipeline presented in Subsection 4.2.1, the goal is to assess how a joint contribution of classification and geolocation can help in identifying real and localized events, disregarding the others.

![Tweet over time](image)

Figure 5.6: Number of tweets per hour in the FLOOD dataset.

5.5.2.1 Classification

A typical system which performs event detection and information extraction uses a classifier to group those tweets related to a particular class. Indeed, since the dataset is obtained simply matching some keywords, there is no prior knowledge
about the content of the tweets nor about their topic.\footnote{For example, some keywords can have different meanings in different contexts.}

Therefore, a classifier (linear Support Vector Machines, SVM) has been trained, using as training sets those provided by \cite{15} related to two important floods happened in Sardegna (November 2013) and in Genova (October 2014). In particular, there are 3 classes:

- "damage": tweets related to a flood and carrying damage information
- "no damage": tweets related to a flood but without any damage information
- "not related": tweets not related to a flood

These two datasets have been integrated by a set of 500 tweets randomly extracted from the day before the first day in the FLOOD dataset and manually annotated with the same 3 classes. Overall the training set includes 1910 tweets, among which 935 of class “damage”, 479 of class “no damage” and 496 of class “not relevant”.

The choice of the training set is due to the fact that the datasets provided by \cite{15} have the same language (Italian) and are related to the same kind of event (floods) as the target dataset, and this should improve the performance \cite{32}.

Among the 500 tweets from the day before the first day in the FLOOD dataset the 77\% has a “not relevant” class and they have been added to model also the “no occurring event” situation, since all the tweets provided by \cite{15} are captured during an event. The training dataset can be considered realistic since it includes only tweets from the past and could be used in a system for automatic real-time event detection and analysis.

As explained in Subsection 2.3.1.1, when the classifier is trained on past events a \textit{domain adaptation} challenge arises. An approach to deal with it is choosing features which are more abstract, to reduce the feature space on one side and to generalize the samples on the other side. For this reason, a \textit{preprocessing} step has been employed, where the occurrences of \textit{urls}, \textit{mentions} and \textit{numbers} are substituted by fixed tokens, “hash” symbols are removed and each token is \textit{stemmed}. Moreover, all the “stopwords” have been removed. As features, all the tokens and bigrams of the tweets are selected (after the preprocessing step), plus a features which accounts for the punctuation used and a features which accounts for the number of hashtags used. Details on these operations in general have been described in Subsection 2.3.1.

The results of the classification task are shown in Table 5.5. It is possible to notice that, since the dataset FLOOD does not focus on a particular event but
5. Experimental evaluation

<table>
<thead>
<tr>
<th>Class</th>
<th># Tweets</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Damage</td>
<td>2022</td>
<td>2.8%</td>
</tr>
<tr>
<td>No damage</td>
<td>6381</td>
<td>8.8%</td>
</tr>
<tr>
<td>Not relevant</td>
<td>64280</td>
<td>88.4%</td>
</tr>
</tbody>
</table>

Table 5.5: Outcome of the classification.

includes also many “normal” days without occurring events, the majority of tweets has been labeled as “not relevant”.

5.5.2.2 Temporal analysis

At this point, only the tweets labeled as “damage” — which constitute the FloodsDamage dataset — are considered. Indeed, in an event detection task they are useful to identify emergency situations and to describe them. In this way, the classifier act as a “filter” to keep only relevant reports. The graph of the damage-labeled tweets over time (#tweets per hour) is shown in Figure 5.7.

It is useful to compare the global amount of tweets to the damage-related tweets over time to qualitatively assess the performance of the classifier and to find out particular situations which can be further analyzed. Figure 5.8 shows the two graphs overlapped, with the graph of the damage-related tweets scaled \(^{12}\) so that the maximum coincide, for ease of comparison.

\(^{12}\)Scale factor = 7.9. The maximum is in correspondence of the day 6/11/2016 for both graphs.
Figure 5.8: Number of tweets per hour vs number of damage-related tweets per hour (scaled to have the same maximum).

The number of tweets for each day are compared to the damage-related tweets in Table 5.6.

Using these comparisons, specific days have been manually analyzed along with an analysis of the events occurred in those days (with the support of online newspapers and the links attached in the tweets themselves). This analysis does not pretend to be exhaustive of the whole dataset but aims to highlight how such a classifier-based filtering is useful in an event detection system and to set up a ground truth about real events occurred in the dataset, which is useful for the following analysis.

In particular, 3 days have been selected to be manually analyzed, based on the qualitative features showed by the graph in Figure 5.8 and the quantitative features showed by the Table 5.6. These days have been marked with a number (1, 2, 3) in the graph and are bold in the table. They are briefly analyzed in the following.

3 - 6/11/2016 This day corresponds to the maximum, both in terms of hourly peak and in terms of overall tweets in the day, both as the global number of tweets and the damage-related tweets and the percentage of damage-related tweets (9.49%). Analyzing the news related to this day, it is evident that
5. Experimental evaluation

<table>
<thead>
<tr>
<th>Month</th>
<th>Day</th>
<th>count global</th>
<th>count damage</th>
<th>% damage</th>
</tr>
</thead>
<tbody>
<tr>
<td>October</td>
<td>14</td>
<td>3866</td>
<td>162</td>
<td>4.190378</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td><strong>2318</strong></td>
<td><strong>145</strong></td>
<td><strong>6.255393</strong></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>2077</td>
<td>53</td>
<td>2.551757</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>2031</td>
<td>29</td>
<td>1.427868</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>1720</td>
<td>45</td>
<td>2.616279</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>1942</td>
<td>36</td>
<td>1.853759</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1928</td>
<td>23</td>
<td>1.192946</td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>1734</td>
<td>64</td>
<td>3.690888</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>1500</td>
<td>18</td>
<td>1.200000</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>1732</td>
<td>18</td>
<td>1.039261</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>1886</td>
<td>33</td>
<td>1.749735</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>1994</td>
<td>48</td>
<td>2.407222</td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>2421</td>
<td>67</td>
<td>2.767451</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>2180</td>
<td>71</td>
<td>3.256881</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>1636</td>
<td>42</td>
<td>2.567237</td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>1940</td>
<td>32</td>
<td>1.649485</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>1791</td>
<td>74</td>
<td>4.131770</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>1859</td>
<td>37</td>
<td>1.990317</td>
</tr>
<tr>
<td>November</td>
<td>1</td>
<td>1739</td>
<td>28</td>
<td>1.610121</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1930</td>
<td>21</td>
<td>1.088083</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2378</td>
<td>47</td>
<td>1.976451</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td><strong>3790</strong></td>
<td><strong>91</strong></td>
<td><strong>2.401055</strong></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2421</td>
<td>22</td>
<td>0.908715</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td><strong>4023</strong></td>
<td><strong>382</strong></td>
<td><strong>9.495401</strong></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>2554</td>
<td>135</td>
<td>5.285826</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>1854</td>
<td>37</td>
<td>1.995685</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>2046</td>
<td>32</td>
<td>1.564027</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>2011</td>
<td>28</td>
<td>1.392342</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>2004</td>
<td>28</td>
<td>1.397206</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>1699</td>
<td>41</td>
<td>2.413184</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>1788</td>
<td>14</td>
<td>0.782998</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>2448</td>
<td>70</td>
<td>2.859477</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>1973</td>
<td>34</td>
<td>1.723264</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>1470</td>
<td>15</td>
<td>1.020408</td>
</tr>
</tbody>
</table>

Table 5.6: For each day in the dataset: number of overall tweets, number of damage-related tweets and percentage of damage-related tweets.
5.5. Geolocation-based pipelines

A real flood occurred (particularly in the areas of Florence and Rome\textsuperscript{13}, with extensive damage and two deaths. In this case, the effect of the filter is not determinant to show the event (since there is a peak also in the raw tweets), but certainly it allows to confirm its relevance in terms of damages and, overall, the relative distance in terms of number of tweets with respect to the other days is largely increased after the filtering, and this can affect positively the event detection phase.

\section*{2 - 4/11/2016} This day is interesting because before the filtering it is one of the major peaks and corresponds to the day with more global tweets after \section*{3}, but, differently from \section*{3}, only a small percentage of tweets (2.40\%) is marked as damage-related and after the filtering the peak is not particularly significant anymore. Indeed, manually analyzing the tweets it is clear that they mostly refer to the commemoration of the 50\textdegree anniversary of an important flood happened in Florence the 4th November 1966\textsuperscript{14}. Therefore, the classifier has been able to distinguish between emergency tweets and commemorative tweets. This example shows the importance of a semantic-related step in performing event detection/analysis: most of the keywords which match an actual flood also match the commemoration of a flood and blindly take all the tweets from the Twitter APIs to perform the subsequent analysis could bring to errors. For example, in this case not only the commemorative tweets match the keywords for a flood, but they are also precisely located (Florence area) and semantically coherent, therefore both semantic-driven and location-driven clustering techniques for event detection (ref. Section \textit{2.2}) would recognize this as an event, bringing a \textit{false positive} as outcome.

\section*{1 - 15/10/2016} This is the only day with a greater peak (in proportion) after the filtering. Moreover, it is the day with the highest percentage of damage-related tweets (6.25\%) and number of damage-related tweets after \section*{3}. Manually analyzing the tweets and the news related to this day it is clear that several real floods with damages occurred in Italy\textsuperscript{15}. However, they are all floods with only local relevance (small-scale events) and therefore the

\begin{itemize}
  \item \textsuperscript{13}http://roma.repubblica.it/cronaca/2016/11/06/news/maltempo_due_morti_a_ladispoli_e_cesano_allagamenti_e_alberiCaduti_nella_capitale-151471826/
  \item \textsuperscript{14}https://en.wikipedia.org/wiki/1966_flood_of_the_Arno
  \item \textsuperscript{15}http://www.meteo.it/giornale/meteo-italia-in-tempo-reale-del-15-ottobre-2016-10227.shtml
\end{itemize}
peak registered is due to the accumulation of several real — but small — events. This example is interesting at least for two reasons:

- The classifier, acting as a filter, is able to emphasize this day (which showed an average peak before the filtering), since it contains many tweets related to actual damages. After the filtering, this day overcomes for example \(^2\) in terms of damage-related tweets. Once again, this shows the usefulness of a semantic step in a system which performs event detection/analysis; in this case to avoid false negatives.

- It shows a limitation of a method which accounts only for the cumulative number of tweets over time to perform event detection (for example through peak detection) since a peak could be due to the contribution of many small events rather than a single important one, as in this case. In this case, an analysis of the tweet themselves is useful to select semantic or geographic bursts. A geographic analysis, performed using the algorithm proposed in Chapter \(^3\) is detailed in the following.

At this point, using the ground truth built in the analysis above, location-based event detection is performed taking as locations the outcome of the proposed text-based algorithm and showing how it can overcome the limitations of a purely time-series based method.

### 5.5.2.3 Location-based event detection

As discussed in Subsection \(^4.2.1\) location-based event detection allows to group together tweets related to near locations. This allows to rule out events which does not have a precise location by design.

Typically this is achieved through density-based clustering methods. Incremental versions of DBSCAN have been employed \(^41, 69\). In the following DBSCAN will be used; this could not be applied for a system which processes tweets in real-time but here the goal is to demonstrate the effectiveness of such a method. As distance metric between two points, the haversine distance\(^16\) is used so that each point is interpreted spatially. Each tweet contributes with a number of points equal to the number of its locations.

First of all, to perform location-based clustering, it is necessary to extract the locations from the tweets.

---

\(^16\)https://en.wikipedia.org/wiki/Haversine_formula
5.5. Geolocation-based pipelines

If only geotags would be taken into account, since in this case the dataset includes only medium/small events in a medium-sized country like Italy, the tweets with an associated location are only a few.

In particular, considering the whole FLOODS dataset, 377 tweets are geotagged. This number is already low considering that there are less than 15 geotagged tweets for each day. However, it is drastically reduced after duplicates are removed and only damage-related tweets are kept: only 4 tweets have an attached geotag. As a side note, manually analyzing the geotagged tweets reveal many of them coming from few bot users that give periodically information (like weather updates).

This situation highlights the importance of text-based geolocation methods to detect medium/small events. If in Subsection 5.5.1 in the context of a big event, it has been shown that a text-based geolocation method allows to overcome some limitations with respect to using only natively geotagged tweets, in this case, in the context of medium/small events in a medium-sized country, even without proceeding with the experiment itself, it is clear that a location-based event detection system cannot rely only on geotagged tweets.

The algorithm described in Chapter 3 has been applied on the FLOODS DAMAGE dataset. The sliding window has been set to 24 hours.

Using the resulting locations, DBSCAN has been applied to each of the 3 days (1, 2, 3) analyzed previously. Overall, 898 locations are disambiguated. Removing location at state or region level, too coarse-grained, 799 remain and are considered in the following.

The parameters of DBSCAN have been set as follows. \( \epsilon = 50 \) km (that is, two points are considered near if their distance is less than 50 km). Different values of \( \text{min-sample} \) (that is, the number of near points necessary to set up a cluster) have been tested: 10, 20, 50, 75.

The results are shown in Table 5.7 and analyzed in the following.

15/10/2016 For \( \text{min-sample} \in \{20, 50, 75\} \), there are no clusters. 2 clusters are created only with \( \text{min-sample} = 10 \). They are shown in Figure 5.9. Manually analyzing the related tweets, it is possible to see that the first one refers to a landslide happened in Sardinia that blocked some streets, the other one is composed by tweets referring to moderate flood damages in the center of Italy (Terni). The first one is composed by few tweets\(^{17}\) therefore

\(^{17}\)Among these few tweets, many have the same authors and almost the same content, but since each tweet have a couple of different hashtags, they were not removed in the deduplication step.
5. Experimental evaluation

<table>
<thead>
<tr>
<th>Day</th>
<th>min_sample = 10</th>
<th>min_sample = 20</th>
</tr>
</thead>
<tbody>
<tr>
<td>15/10/2016</td>
<td># cluster</td>
<td>% noise</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>51%</td>
</tr>
<tr>
<td>4/11/2016</td>
<td>1</td>
<td>44%</td>
</tr>
<tr>
<td>6/11/2016</td>
<td>3</td>
<td>3%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Day</th>
<th>min_sample = 50</th>
<th>min_sample = 75</th>
</tr>
</thead>
<tbody>
<tr>
<td>15/10/2016</td>
<td># cluster</td>
<td>% noise</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>4/11/2016</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>6/11/2016</td>
<td>2</td>
<td>17%</td>
</tr>
</tbody>
</table>

Table 5.7: Outcome of DBSCAN location-based clustering for each of the three days according to different values of min_sample. For each value of min_sample the number of formed clusters and the percentage of noise (that is non-clustered tweets) is reported.

increasing min_sample the cluster does not exist anymore. The second one is composed by the locations related to a small flood and, even if many other tweets related to floods exist in the center of Italy (since, as analyzed previously, this day is characterized by many distributed small floods) the locations are not so dense and are not recognized as a cluster increasing min_sample — simply because they are indeed many small events, not an unique one. This is confirmed also by the fact that, even when 2 clusters are detected, the majority of the locations are not clustered (noise = 51%). In a real system interested at least in medium-sized events (at which correspond a high min_sample), no events will be detected using this location-based approach.

2 4/11/2016 For min_sample ∈ {20, 50, 75}, there are no clusters. 1 cluster is created only with min_sample = 10. It is around the area of Florence, clearly representative of the commemorative tweets of the day which have not be filtered out by the classifier. Increasing min_sample the cluster disappear because the majority of commemorative tweets have been filtered out: only 27 coordinates are extracted in the day, and many are not Florence-related, since even when the cluster is formed the 44% of tweets are excluded. Also in this case, in a real setting with a high min_sample the commemorative event would not be detected.
5.5. Geolocation-based pipelines

Figure 5.9: The clusters formed the day 15/10/2016 with \textit{min\_sample} = 10. They are circled in figure only for ease of viewing.

6/11/2016 In this case, clusters are formed with any value of \textit{min\_sample}, as shown in Table 5.7. For lower values of \textit{min\_sample} (\{10, 20\}) 3 clusters are formed (see Figure 5.10): near to the two clusters related to the 2 main events of the day (Rome and Florence), there is a third cluster which includes reports of minor damages in Arezzo and Siena. However, for high values of \textit{min\_sample} (\{50, 75\}) only the 2 clusters related to Rome and Florence are formed (see Figure 5.11). This result is in line with the ground truth built previously (ref. 5.5.2.2), where Florence and Rome were identified as the main floods of the day in terms of damages and deaths. Moreover, notice that these 2 clusters are the only ones identified among the 3 analyzed days with a high \textit{min\_sample}, that is, setting a threshold to keep at least medium-sized events. Finally, notice that the amount of not clustered tweets is negligible with \textit{min\_sample} \in \{10, 20\} (noise = 3\%) and is anyway low with \textit{min\_sample} \in \{50, 75\} (noise = 17\%). This means that this day is characterized by a more “dense” distribution of tweets with respect to the others.
Figure 5.10: The clusters formed the day 6/11/2016 with \textit{min\_sample} = 10 and \textit{min\_sample} = 20. They are circled in figure only for ease of viewing.
Figure 5.11: The clusters formed the day 6/11/2016 with \( \text{min\_sample} = 50 \) and \( \text{min\_sample} = 75 \). They are circled in figure only for ease of viewing.
5. Experimental evaluation

5.5.2.4 Discussion

This case study has demonstrated how the processing pipeline designed in Subsection 4.2.1 is able to overcome some of the typical challenges of event detection thanks to a joint contribution of classification and location-based clustering assisted by geolocation.

2 is an example of event that cannot be filtered out simply through the keywords used for acquiring the tweets, because many of the words used referring to the commemoration of an event obviously coincide with the words typically used during an actual event of the same type. Moreover, 2 includes tweets that does not only share the same topic, but also refer to the same location, therefore any kind of clustering-based method for event detection (included location-based event detection) would detect 2 as an actual event. Therefore, classification is needed to interpret semantically the tweets, for example to understand if they refer to actual damages. The classification step has filtered out the majority of the tweets related to the commemoration, so that the related cluster does not exist anymore with any reasonable parameter choice.

1 is not an event of particular importance, but instead includes many very small events. Classification is not enough to correctly interpret it, since it only allows to confirm that there exist a consistent number of damage-related tweets. However, thanks to location-based clustering it is possible to see that they are not enough “dense” from a geographical point of view to form a proper event. Notice that these tweets share similar topics, therefore would be challenging to cluster them from that point of view 18, and after the classification step the number of geotagged tweets is irrelevant and therefore location-based event detection would be impossible without a geolocation processing step.

Finally, 3 includes two events which satisfy both the conditions: they have a high number of damage-related tweets and are spatially dense. They would be detected as events by a real system, and indeed they were recognized as real significant events during manual analysis. The geolocation processing step is necessary because the number of natively geotagged tweets would be irrelevant after the classification-based filtering step, proving itself as an “enabler” for the pipeline.

18That is, recognizing that they are actual distinct events.
5.5.3 Image analysis

Images represent one of the resources that is possible to extract from social media and can have a huge impact on added situational awareness when analyzed.

In Subsection 5.4.3 an example of a useful image showing a damaged street has been reported with a discussing on how it is possible to correctly localize it using its text thanks to the proposed algorithm.

Indeed, exactly as any other kind of information, during a crisis scenario images are useful only when associated (automatically or manually) to a location.

However, near to a small percentage of natively geotagged tweets there is also a modest percentage of tweets with an associated image, bringing the overall number of tweets both natively geotagged and with an associated image to an even smaller percentage.\textsuperscript{19} For example, in the AMATRICE dataset $\approx 0.27\%$ of tweets are natively geotagged and $\approx 15\%$ of tweets have an associated image. Only 29 (0.022\%) images are attached to natively geotagged tweets. Moreover, considering that not all the images will be useful in terms of added situational awareness and can be duplicates, the set of unique images geotagged and useful could include only few elements.

Therefore, this is another case in which a geolocation processing phase could indirectly positively affect the outcome of a crisis-related application thanks to its ability to assign a location to more tweets and then potentially to more useful images, as explained in Subsection 4.2.3.

This situation is studied in the following on the AMATRICE dataset.

First of all it has been searched in the dataset for keywords related to potentially useful images (that is, images of precise locations) like “road”, “street”, etc. The complete keyword list and extraction procedure is detailed in \textsuperscript{19}.

The resulting tweets include 973 images. Among these no image is associated to a natively geotagged tweet. This highlights the necessity of a geolocation step.

Among the related tweets, the proposed text-based geolocation algorithm is able to assign at least a location to 360 tweets (37\%) (discarding states and regions because too coarse-grained).

To assess the performance on unique and useful images, the images have been further processed:

- First of all, duplicate images have been removed. These should include not only identical images but also different versions of the same original photos.

\textsuperscript{19}Images lose all metadata, including their geographical coordinates, when stored by Twitter, therefore it is not possible to use the associated coordinates to localize them.
5. Experimental evaluation

in terms of dimensions, resolution and colors. For this purpose, the perceptual hashing algorithm pHash has been employed. It has been preferred a conservative approach which minimizes the risk of false positives, thus leaving some duplicate images in the result. When more images were available, the highest resolution one has been chosen. This step has reduced the 973 images initially considered to 783 images.

- The resulting images have been manually annotated to identify the useful ones. Since the idea is to find images useful for the emergency management task, the following criteria has been adopted:
  - are considered useful the images which show damaged or blocked streets and roads, damage to buildings and population or aerial images;
  - are excluded the images which focus on the rescue operations, minor damages, taken inside, focusing only on a small detail, with a resolution too low to be understandable or photos of other photos.

The image annotation task has taken into consideration only the images and not the text or any metadata of the tweets. At the end, 111 images were marked as useful.

Among the tweets associated to the resulting 111 useful and unique images, the algorithm is able to assign a location to 54 (48%).

The results are summarized in Table 5.8.

<table>
<thead>
<tr>
<th>Images</th>
<th>Geolocated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location-related images (973)</td>
<td>360 (37%)</td>
</tr>
<tr>
<td>Unique and useful images (111)</td>
<td>54 (48%)</td>
</tr>
</tbody>
</table>

Table 5.8: The amount of geolocated images considering all the extracted images and only the useful and unique ones.

5.5.3.1 Discussion

A first important result is that, given the analysis reported in Sections 5.3 and 5.4, it’s known that the algorithm has an almost-absolute precision in identifying the locations and accuracy in disambiguating them, therefore the images geolocated by the algorithm will be mostly correctly geolocated.

\[\text{https://github.com/JohannesBuchner/imagehash}\]
An observation arises considering the percentage of tweets with disambiguated locations. It was reported (ref. Subsection 5.4.3) that on the entire dataset ≈ 20% of tweets are geolocated by the algorithm. This percentage increases significantly considering only the tweets subject of the previous analysis, and even more considering only those with a useful image (Table 5.8). This could be the result of a correlation between tweets with a useful attached image and tweets with a location specified in the text (that is, who posts a useful image is more prone to specifying the related location in the message content).

An important limitation of the algorithm in this context is that, even if state and region level locations have been discarded, the resulting locations are in any case too coarse grained to be directly associated to the subject of the images. On one side this is due to the fact that the used gazetteer (GeoNames) does not have many names of streets, roads, buildings etc., and on the other side to the fact that many images are related to unnamed locations (like private buildings or areas). If the first limitation could be overcome by using a gazetteer which also includes fine-grained locations, the second limitation is intrinsic of any algorithm which performs geolocation based on the toponyms reported in the text.

Nonetheless, the locations extracted by the algorithm could be useful in various ways: for example, in a system which gives the images to the crowd to be precisely geolocated, it is possible to deliver the images related to a city directly to the inhabitants of that city speeding up the operation, or they could be used as input of another system that, knowing that a certain image belongs to a certain city, automatically assigns it to the building/street subject of the image.
Chapter 6

Conclusion and future work

In this thesis was presented a new algorithm able to exploit the social networks naturally occurring during emergencies to automatically geolocate individual messages, showing how this enables and enhances other tasks in the emergency management area.

The proposed algorithm has been evaluated demonstrating an high precision in identifying the locations and accuracy in disambiguating them, especially focusing on the locations affected by the emergency.

Moreover, it has been evaluated in the context of typical applications in the emergency management area through several case studies: event detection, situational awareness support and image analysis. For each of these tasks the contributions of the proposed algorithm have been quantified and compared to the use of machine-readable location information (metadata) only. In particular, the locations identified by the algorithm have been able to overcome the limitations of machine-readable locations in terms of volume, velocity and location type, proving themselves useful in enhancing situational awareness quicker and clearer, identifying events — especially small-medium events — with a location-driven approach and attaching locations to about half of the extracted damage-related images.

In conclusion, this thesis has demonstrated the effectiveness of behavioral social networks as a resource for the message geolocation task on social media in the context of emergency management applications.

Future developments could further deepen the use of behavioral social networks for message geolocation.

The proposed algorithm can have many variations, especially in terms of the relationships considered (there is a line of research studying the different social networks characterizing social media) and the subject of future research could be
6. Conclusion and future work

to systematically investigate how different relationships impact on the geolocation performance. Other interesting variations concern the use of a weighted graph to give a different importance to different relationships among messages and the introduction of a “partially disambiguated” state for messages to account for situations where some locations have been disambiguated but others still need a context. Moreover, variations to the sliding window model could be taken into account, together with an explicit role of time to set up relationships.

In this work it has been demonstrated that the proposed algorithm is able to operate online and under certain constraints can process any new message in a bounded time on average; the subject of future research could be to precisely assess the time and the resources required, linking also this aspect to the different types of relationships considered, in particular taking into account the approximations which allow to give up some precision in order to speed up the comparisons among messages (like hashing), introducing real-time processing as new dimension.

Describing the algorithm, it has been shown that it can be applied also to location inference, not only identification and disambiguation. However, inference evaluation is more difficult since it is hard to build a ground truth about locations when there are no explicit mentions, and for this reason its evaluation has not been carried out in this thesis. Using a group of independent annotators and a ground truth built on the basis of their agreement, it will be possible to precisely evaluate also location inference in future. Moreover, the current implementation for location inference is naïve and other inference functions described in the literature could be implemented in the future.

The gazetteer used in the current implementation, GeoNames, has some limitations, especially in terms of the granularity of its locations. Indeed, it has few names of streets, roads and buildings, but these places are very useful in the emergency management context. The proposed algorithm does not make distinction about different kinds of locations and could theoretically identify and disambiguate roads and streets, but another gazetteer with more resources about them should be used, like OpenStreetMap. Its integration could be the subject of future work.
Bibliography


[56] Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story
detection with application to twitter. In Human Language Technologies: The
2010 Annual Conference of the North American Chapter of the Association
for Computational Linguistics, pages 181–189. Association for Computational

[57] Bruno Pouliquen, Marco Kimler, Ralf Steinberger, Camelia Ignat, Tamara
Oellinger, Ken Blackler, Flavio Fuart, Wajdi Zaghouani, Anna Widiger, Ann-

[58] Afshin Rahimi, Duy Vu, Trevor Cohn, and Timothy Baldwin. Exploiting
text and network context for geolocation of social media users. arXiv preprint

[59] Timo Reuter and Philipp Cimiano. Event-based classification of social me-
dia streams. In Proceedings of the 2nd ACM International Conference on

[60] Alan Ritter, Sam Clark, Oren Etzioni, et al. Named entity recognition in
Methods in Natural Language Processing, pages 1524–1534. Association for

Where’s@ wally?: a classification approach to geolocating users based on
their social ties. In Proceedings of the 24th ACM Conference on Hypertext

[62] Koustav Rudra, Siddhartha Banerjee, Niloy Ganguly, Pawan Goyal, Muham-
mad Imran, and Prasenjit Mitra. Summarizing situational tweets in crisis
scenario. In Proceedings of the 27th ACM Conference on Hypertext and Social

[63] Adam Sadilek, Henry Kautz, and Jeffrey P Bigham. Finding your friends and
following them to where you are. In Proceedings of the fifth ACM international

[64] Tatsuhiro Sakai, Keiichi Tamura, and Hajime Kitakami. A new density-based
spatial clustering algorithm for extracting attractive local regions in georef-


[66] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. IEEE Transactions on Knowledge and Data Engineering, 27(2):443–460, 2015.


