Fabio da Costa Albuquerque

Environment changes detection: A proactive system to monitor moving objects

Dissertação de Mestrado

Dissertation presented to the Programa de Pós-Graduação em Informática of the Departamento de Informática, PUC-Rio as partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Prof. Marco Antonio Casanova

Rio de Janeiro

December 2012

Pontifícia Universidade Católica do Rio de Janeiro

Fabio da Costa Albuquerque

Environment changes detection: A proactive system to monitor moving objects

Dissertation presented to the Programa de Pós-Graduação em Informática do Departamento de Informática do Centro Técnico e Científico da PUC-Rio, as partial fullfilment of the requirements for the degree of Mestre.

Rio de Janeiro, December 18th, 2012

Fabio da Costa Albuquerque

graduated in Computer Technology at Associação Educacional São Paulo Apóstolo, known as UniverCidade (2007). He has been acting in applied research and software engineering, focused on geographic information systems, since 2006 for the Tecgraf PUC-Rio lab.

Bibliographic data

Albuquerque, Fabio da Costa

Environment changes detection: a proactive system to monitor moving objects / Fabio da Costa Albuquerque; advisor: Marco Antonio Casanova. – 2012.
67 f. : il. (color.) ; 30 cm

Dissertação (mestrado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2012.
Inclui bibliografia

1. Informática – Teses. 2. Monitoramento proativo. 3. Aprendizado de máquina. 4. Extração de informação. 5. Sistema de informação geográfica. I. Casanova, Marco Antonio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Acknowledgements

To my advisor Marco Antonio Casanova, for sharing his knowledge throughout the development of this work and for always motivating and collaborating with scientific production.

To Professor Ruy Luiz Milidiú, for his collaboration during the initial stages of this project.

To the Tecgraf laboratory and PUC-Rio university, for providing the resources and infrastructure that made this work possible.

To my parents and siblings, for the unconditional support in every moment of my life.

To my wife Roberta, for her patience and understanding during these last years

Resumo

Albuquerque, Fabio da Costa; Casanova, Marco Antonio. **Detecção de mudanças no ambiente: Um sistema proativo para monitorar objetos móveis.** Rio de Janeiro, 2012. 67p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Sistemas de posicionamento, combinados com tecnologias de comunicação de baixo custo, abrem possibilidades interessantes para implementar aplicações em tempo real que monitoram objetos móveis e que apoiam sistemas de tomada de decisão. Inicialmente, esta dissertação discute requisitos básicos para aplicações proativas de monitoramento em tempo real. Em seguida, propõe uma arquitetura para aplicações proativas que monitoram objetos móveis, explorando a semântica da trajetória e a dinâmica do ambiente. Por fim, fornece um exemplo sobre como uma aplicação que monitora uma frota de caminhões pode se tornar proativa, utilizando notícias sobre condições da malha viária, a partir da publicação de dados em texto não estruturado através da Internet. A dissertação descreve como estruturar e georreferenciar as notícias, utilizando serviços de geocodificação.

Palavras-chave

Monitoramento proativo; notícias relacionadas ao tráfego; aprendizado de máquina; aprendizado supervisionado; extração de informação; sistemas de informação geográfica.

Abstract

Albuquerque, Fabio da Costa; Casanova, Marco Antonio (Advisor). **Environment changes detection: A proactive system to monitor moving objects.** Rio de Janeiro, 2012. 67p. MSc. Dissertation - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Positioning systems, combined with inexpensive communication technologies, open interesting possibilities to implement real-time applications that monitor moving objects and that support decision making. This dissertation first discusses basic requirements for proactive real-time monitoring applications. Then, it proposes an architecture to deploy applications that monitor moving objects, are pro-active, explore trajectory semantics and are sensitive to environment dynamics. Lastly, this dissertation provides an example of how an application that monitors a fleet of trucks can become proactive, using unstructured text information available on Internet focused on road conditions change. The dissertation describes how to structure and geo-reference the text, using available geocoding services.

Keywords

Proactive monitoring; traffic related news; machine learning; supervised learning; information extraction; geographic information systems.

Sumário

Lista de figuras

Lista de tabelas

1 Introduction

Positioning systems, combined with inexpensive communication technologies, open interesting possibilities to implement real-time applications that monitor moving objects and that support decision making. An application to monitor a fleet of tank trucks that distribute fuel to gas stations in an urban environment is a good example. Every trip is carefully planned to follow pre-defined routes, avoiding sensitive areas (such as school areas) and periods of the day or routes where the transportation of dangerous cargo is banned. Each trip is monitored to detect unplanned stops (for security reasons), as well as other incidents, and to pro-actively re-route the truck in case of traffic accidents and other events that might cause delays. If the truck becomes involved in an accident, an appropriate emergency plan is triggered to mitigate the effects of the accident and to try to prevent further problems.

Such monitoring applications are classified according to different perspectives. The application may use *trajectory semantics*, such as stopping at a point of interest, or the application may use just *raw trajectory data*, such as speed and direction. Trajectory semantics is in fact a useful source of information in the context of decision support systems.

A *reactive application* uses just the past behavior of the objects, as opposed to a *proactive application* that features models of the predicted (future) behavior of the objects. A proactive application will typically analyze known (present or future) facts to detect their potential impact on the future behavior of an object and perhaps suggest alternative actions.

Finally, the application may be *sensitive to environment dynamics*, meaning that it monitors the current state of the environment (or even estimates future states of the environment) where the object is moving to base its decisions. Environmental facts are considered when they directly affect the moving object behavior. By contrast, the application may be *insensitive to environment dynamics*, in the sense that it has just a static model of the environment (such as a road map) where the object is moving.

This dissertation focuses on *proactive moving objects monitoring applications* or, simply, *proactive monitoring application.* Such applications: (1) monitor moving objects; (2) are proactive; (3) explore trajectory semantics; and (4) are sensitive to environment dynamics.

The dissertation first lists the requirements and proposes an architecture for proactive monitoring applications. To achieve proactive behavior and anticipate the occurrence of unexpected facts, the proposed architecture includes models of the processes behind the moving objects. To monitor moving objects, the architecture includes support for real-time trajectory data stream processing. Finally, to account for trajectory semantics and support sensitivity to environment dynamics, the architecture features additional data sources, classified as (geospatial) static structured data sources and dynamic structured data sources.

Then, the dissertation describes an application that monitors a fleet of trucks and outlines how to make it proactive. A prototype was developed to enhance the current implementation of the application. The prototype processes tweets to extract and georeference traffic-related facts about road conditions that may affect (future) truck trips. Since the process of extracting traffic-related events and conditions from tweets and georeference them is not a simple task, this dissertation covers these topics with a higher level of detail.

This dissertation is organized as follows. Chapter 2 describes related work. Chapter 3 describes the requirements and the proposed architecture for proactive monitoring applications Chapter 4 describes an application that monitors a fleet of trucks. Chapters 5 and 6, which are the main contribution of this dissertation, detail how to extract and georeference traffic-related facts from tweets. Chapter 6 contains conclusion and future work.

2
Related work

The first aspect to address in moving objects monitoring is the underlying technologies for automatically acquiring data about the object location. Systems for automatic vehicle location (AVL) and mobile communications can be combined to support Intelligent Transportation Systems (ITS). The USA Department of Transportation outlines these resources for commercial vehicle fleet management.

Location is usually provided by Global Navigation Satellite Systems (GNSS), such as GPS and Glonass, as exposed by Tsakiri et al. (1998) and by Prakash and Kulkarni (2003). Examples of references for spatial queries are provided by Pfoser and Jensen (2001), Chon, Agrawal and Abbadi (2002), Güting (2008) and Güting, de Almeida and Ding (2006).

Trajectory data processing is in fact a special case of data stream processing. Aggrawal (2007) describes how to fetch models and algorithms to handle classification, change detection, sliding window computation, joining, forecasting and distributed mining in data streams.

Certain characteristics of object movement may be inferred from the sequence of object locations. As examples, Siqueira and Bogorny (2011) investigate how to identify if a moving object is persecuting another moving object, and Alvares et al. (2011), which address how to detect if moving objects typically avoid some geographic place, and Moreno, Times, Renso and Bogorny (2010), which use raw trajectory data to detect when the object stops.

Proactive computing is investigated in Tennenhouse (2000), which advocates a paradigm shift from human-centered to human-supervised computation. In his perspective, a system to be proactive must: (1) have a direct connection with the real world; (2) be able to execute actions in response to external stimuli; (3) execute actions faster than the human response. In other words, a system with proactive behavior must detect interesting situations before they happen and must be able to handle such situations without human supervision.

Proactivity is two-fold: situations may be detected from past behavior of the object or from external agents that affect the application.

Magoutas, Mentzas and Apostolou (2011) propose a proactive application that performs intelligent energy distribution to end-use customers, predicting the energy consumption based on historical data, customers information and events from various heterogeneous sources (e.g. social networks, customer GPS car system, etc).

Santos and Moreira (2010) propose an input for proactive computing by predicting the next step of moving objects based in its current location and road data. Previous moving object data is not used. The success of prediction may vary according to the scenario and variables. Other approach to proactivity is based on the extraction of relevant facts that potentially affect the future behavior of moving objects, as certain Web applications generate data streams from which such facts may be extracted.

The architecture proposed by Fritz et al. (2010) adopts an a priori crawling of the Web in search of relevant data about objects belonging to non-trivial categories, such as potential emergency shelters and hospitals. Missing location references are computed from addresses by using the Google Maps Geocoding Service.

Sakaki, Okazaki and Matsuo (2010) discuss the semantic analysis of tweets. The authors analyze tweets with respect to the relevance of their contents and to spatio-temporal occurrences. The analysis of spatial aspects considers only the registered locations (tweets from GPS enabled devices). The model also supposes that the registered location is near to the reported event location. The authors explore, as use cases, the problem of estimating the location of an earthquake center and the problem of estimating the trajectory of a typhoon.

Earle, Bowden and Guy (2011) introduce a model for earthquake detection. Event detection is based on the increasing rate of earthquakes tweets. Tests were performed over downloaded tweets containing the words earthquake, and its equivalent in other languages, that were collected during four months. Locations for tweets were also retrieved either by GPS register or by using the Google Maps Geocoding Service to infer the latitude and longitude from the static location string found in a user profile. Because geocoding for locations is not as precise as for addresses, the location statement of a user profile does not provide accurate results.

Mac Eachren et al. (2011) handle missing location registers by considering the location stored in user profiles, place references extracted from tweet content, and hashtags that are associated with places. Predefined keywords and phrases are selected to refine queries on tweets. The resulting tweets are then analyzed and the locations that are extracted are georeferenced using GeoNames. As an example of an application, the authors analyze tweets to outline the role of one church to organize relief efforts to help victims of the Haitian earthquake.

As an example of RSS feeds processing, Chen et al. (2007) use a rule-based tagger that extracts potential place names from a given text and matches them against a location database.

Carvalho, Sarmento and Rossetti (2010) classify tweets indicating the occurrence of messages related to traffic, using machine learning techniques over a dataset with traffic-related and non-related messages.

Structuring raw text data and extracting relevant information is not a trivial task. The Locus system [Souza *et al.*, 2004], an urban spatial finder, has an advanced search feature with a georeferencing objective similar to ours, although with a different implementation. It allows searches with “where” and “what” inputs, similarly to our reference approach. Borges *et al*. (2007) use predefined patterns to extract addresses from Web pages using a set of regular expressions.

In our case, however, using a set of regular expressions, such as an address, a place, a neighborhood or a city to extract locations from raw text would not be very effective, for reasons discussed in Chapters 4 and 5.

3
Proposed architecture for proactive monitoring

3.1.
Basic requirements for proactive monitoring

To achieve proactive monitoring of moving objects, the system must:

1. Model the behavior of moving objects and monitor the current state of objects.
2. Model the environment where objects move and monitor the current state of the environment.
3. Detect environment changes that may affect the future behavior of the moving objects and adjust their behavior to the changes.

3.1.1.
Modeling moving objects behavior

It is required that the system includes a model of the behavior of moving objects and that the system is able to monitor the current state of objects.

Consider a moving object *M* as the manifestation of a process *P*, modeling the behavior of *M* means to model process *P*. The accuracy of the model naturally depends on the requirements of the application. For the purposes of the applications in question, it is required that the system supports modeling process *P* as a workflow W_M.

The workflow language must support certain features to be useful for monitoring moving objects. Each *step p* of W_M is associated with an *action* A_p, a *location* O_p and a *time interval* $[B_p,E_P]$, where B_p is the *beginning* of the interval and E_P is the *end*.

Among the possible actions, must be included: the *moving* from O_p to a location D_p, called the *destination* of *p*, through a *route* R_p, within the time interval $[B_p, E_P]$; the route R_p must start on O_p and end on D_p. Also, the location and the time interval of any step *q* that follows *p* in W_M must be consistent with those of *p*: the destination of *p* must be the location of *q* and the end of the interval of *p* must be the beginning of that of *q*.

Let W_M be the workflow associated with a moving object *M*. To monitor *M*, the system must be able to detect the current step of the execution of W_M, including the current location of *M*. All future decisions will be based on the current location and on the next steps of W_M.

Finally, the system must be able to assess whether the execution of the next steps of W_M is still feasible. Otherwise, an alternative workflow is proposed in order to achieve the same effects of the remaining steps of W_M, considering the available routes and resources.

3.1.2. **Modeling the environment**

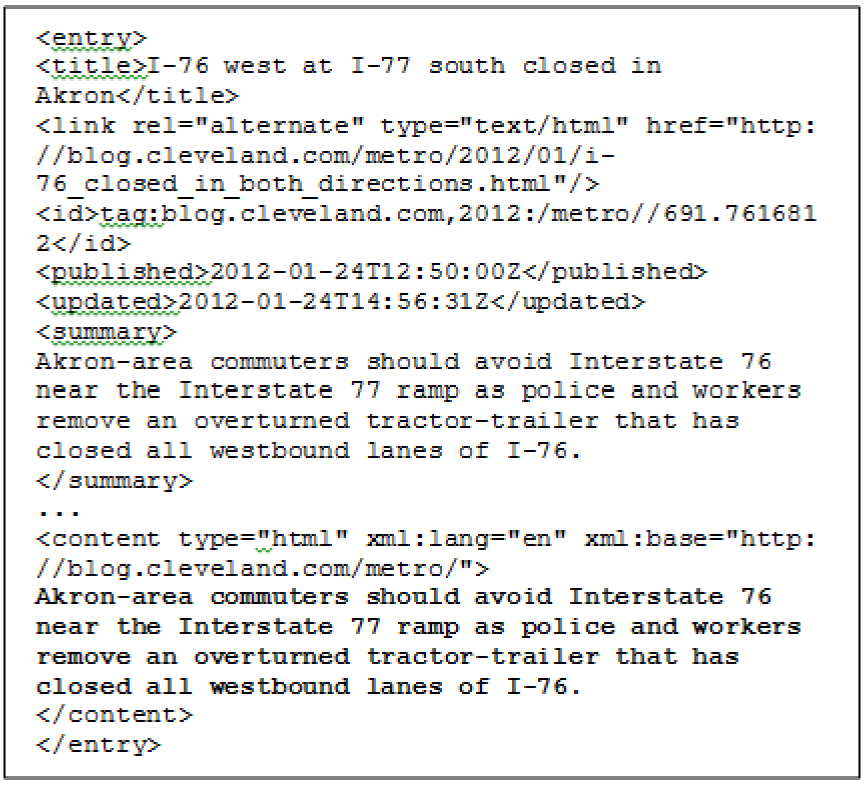
As requirement, the system must include a model of the environment where objects move and that the system is able to monitor the current state of the environment.

Typically, this means that system must include a collection of data sources, classified as *(geospatial) static structured data sources (SSD sources)* and *dynamic structured data sources (DSD sources)*.

An *SSD* source contains data about geospatial features with fixed location and low rate of update, such as roads, water courses and buildings. Hence, possible *SSD* sources are (national) Spatial Data Infrastructures (*SDI*) nodes, Web services able to handle geospatial data, georeferenced linked data (accessed through SPARQL queries) and Volunteered Geographic Information (*VGI*). *VGI* is a relevant data source when the methodology for data collection is ruled by proper specifications, or in the absence of more sophisticated data. Otherwise, raw data need to be processed to filter discrepancies and estimate missing values. The same is valid for sensor networks and user feedbacks.

A *DSD* source delivers data about dynamic phenomena that temporarily modify static geospatial features. The update rate of a *DSD* source must be higher than that of a *SSD* source. A *DSD* source is therefore essential to monitor the current state of the environment. The following *DSD* sources can be listed as candidates: tweets, *RSS* and *geoRSS* (http://georss.org/) feeds and Open GeoSMS (OGC, 2012). Figure 2 illustrates a RSS feed containing relevant data in the *content* tag.

Finally, to select a set of data sources from those available, the system should preferably evaluate them according to their *quality of services* (Claro, Albers and Hao, 2006): *cost*, *time*, *availability* and *reputation*. The system may also evaluate the data sources with respect to their trustworthiness, estimated based on their spatial coverage, data completeness, provenance and lifetime (Barbosa and Casanova, 2011).



```
<entry>
<title>I-76 west at I-77 south closed in
Akron</title>
<link rel="alternate" type="text/html" href="http:
//blog.cleveland.com/metro/2012/01/i-
76_closed_in_both_directions.html"/>
<id>tag:blog.cleveland.com,2012:/metro//691.761681
2</id>
<published>2012-01-24T12:50:00Z</published>
<updated>2012-01-24T14:56:31Z</updated>
<summary>
Akron-area commuters should avoid Interstate 76
near the Interstate 77 ramp as police and workers
remove an overturned tractor-trailer that has
closed all westbound lanes of I-76.
</summary>
...
<content type="html" xml:lang="en" xml:base="http:
//blog.cleveland.com/metro/">
Akron-area commuters should avoid Interstate 76
near the Interstate 77 ramp as police and workers
remove an overturned tractor-trailer that has
closed all westbound lanes of I-76.
</content>
</entry>
```

Figure 1 Example of unstructured data within RSS feed (http://blog.cleveland.com/traffic/atom.xml)

3.1.3.
Future behavior of moving objects

It is required that the system is able to detect environment changes that may affect the future behavior of a moving object and that it adjusts the behavior of the objects to the changes. The environment changes are associated to events or facts of interest that affect the environment.

Recall from Section 3.1.1 the modeled process associated with a moving object M as a workflow W_M, where each step p of W_M is associated with an action A_p, which has a location and a time interval. Also recall that among the possible actions, the *moving* from location O_p to location D_p through a *route* R_p, within a time interval was included.

Therefore, when simulating the execution of W_M, the system may detect potential problems with the execution of future steps due to a change in the state of the moving object or in the state of the environment, which depends on the application, or due to the spatio-temporal specification of future steps, which is application-independent.

The system must then analyze data retrieved from both static and dynamic structured data sources (defining the state of the environment), data about the moving object (defining its state) and data associated with future steps to detect changes that may affect the current workflow execution. In particular, the system must be able to analyze the available data to detect facts that affect routes that the moving object may pass.

When a change in the environment affects future steps, the system must assess the feasibility of the workflow and suggest alternatives. The rules that govern this process are application-dependent. In general, the following are feasible alternatives: the workflow execution may be delayed until the environment returns to a state where execution can be retaken; the workflow may be replaced by another workflow that better fits the current environment state; the workflow may be adapted according to some redefined strategy (Vieira, Casanova and Ferrão, 2004).

3.2. An architecture for proactive monitoring applications

3.2.1. Overview of the architecture

Figure 1 illustrates the proposed architecture. The Proactive Central Monitor (PCM) is the core component that, as the name implies, coordinates the other components to pro-actively monitor moving objects. The Planning Manager (PM) stores and controls the workflows that model the behavior of the moving objects. The Application Databases contain auxiliary data such as names and addresses of customers, the road network, etc. The Moving Objects Monitor (MOM) sends to the PCM the structured data stream containing information relative to the real-time monitoring of moving objects: position, trajectory semantic data (i.e., interpreted trajectory data) and other signals from moving objects. The Mediators facilitate access to external data sources.

The Mediators, the Planning Man*ager* and the *Proactive Central Monitor* are explained in more detail in the next sections.

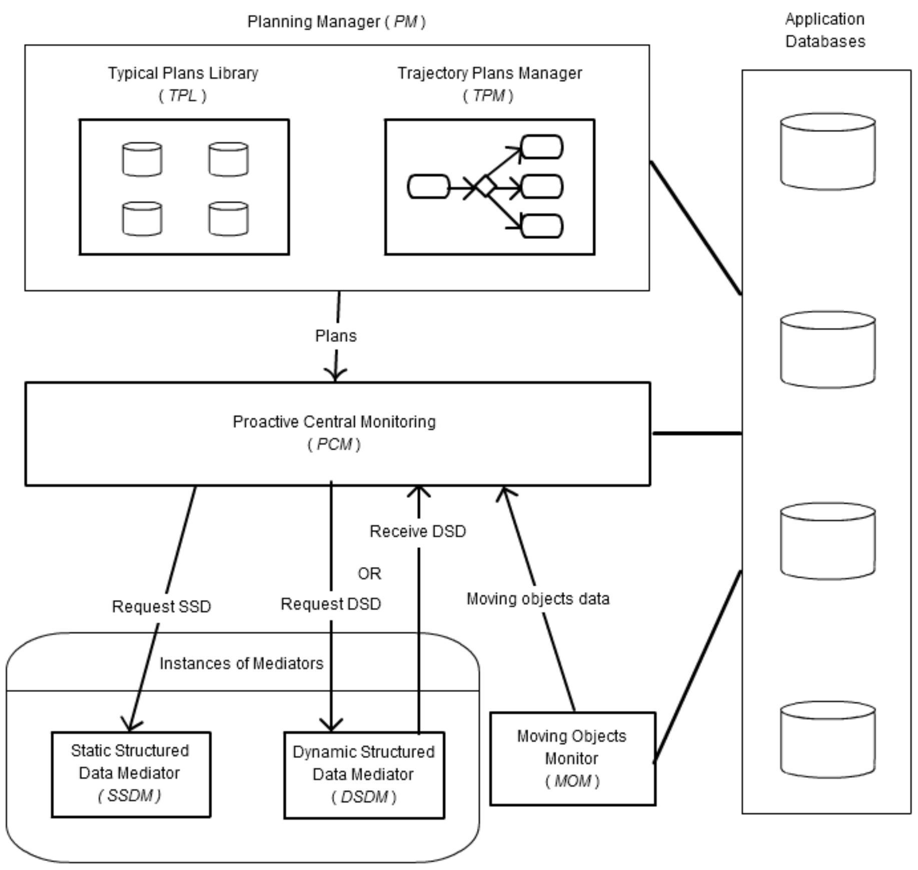

Planning Manager (*PM*)
Application Databases
Typical Plans Library (*TPL*)
Trajectory Plans Manager (*TPM*)
Plans
Proactive Central Monitoring (*PCM*)
Receive DSD
OR
Request SSD
Request DSD
Moving objects data
Instances of Mediators
Static Structured Data Mediator (*SSDM*)
Dynamic Structured Data Mediator (*DSDM*)
Moving Objects Monitor (*MOM*)

Figure 2 General view of architecture proposal

3.2.2. Structured Data Mediators

The PCM receives data from the Dynamic Structured Data Mediator (DSDM) and the Static Structured Data Mediator (SSDM), designed as suggested in Bayegan (1998). These mediators are devoted to process the queries that the PCM submits. They access wrappers that encapsulate the data sources.

The design of the mediators and wrappers may follow two basic strategies. The first strategy transfers to the wrappers most of the query processing tasks and indicates when specialized wrappers are available. The second strategy defines that the mediator is responsible for post-processing the results returned by the wrappers, and is indicated when specialized wrappers are not available. In either case, the mediators must contain application-specific rules to classify the data returned by the wrappers as relevant or irrelevant, and discard irrelevant data.

Finally, the mediators must transform raw relevant data into structured data to be sent to the PCM.

3.2.3. Planning Manager

The Planning Manager (PM) contains a Trajectory Planning Manager (TPM) and a Typical Plans Library (TPL).

The TPM is responsible for managing the workflows that model the behavior of the moving objects.

The TPL is designed to handle emergencies and other unanticipated events related to the application (e.g. broken vehicle and load spill). The TPL essentially is a library of typical workflows, designed to mitigate emergencies.

3.2.4. Proactive Central Monitor

The Proactive Central Monitor (PCM) has five major objectives, as discussed in detail in Section 3.1:

1. Monitor the current state of moving objects.
2. Monitor the current state of the environment.
3. Detect environment changes that may affect the future behavior of the moving objects.
4. Adjust the behavior of the objects to the changes.
5. Detect emergencies and invoke the TPL for the proper actions.

To achieve the first and the last objectives, the PCM combines data from the MOM with workflow execution control. The second objective is achieved by processing data from the SSDM and the DSDM. The third and fourth objectives require workflow execution simulation.

Finally, a few design issues that affect the implementation of the PCM are listed:

- Monitoring the current state of the environment should be optimized, possibly by ordering data received from the DSDM by time stamp.
- If the number of monitored objects is large, processing the data from the SSDM may become a bottleneck. Caching such data may be required in this case.

- An implementation strategy based on small, reusable modules that meet the needs of several applications may be desirable, as the number of applications grow.

3.2.5. Moving Objects Monitor

Section 3.2.1 describes, in abstract terms, how to model the behavior of the monitored moving objects. In order to use the proposed model, certain information about what the vehicle is doing (action) and where it is (state) is necessary. Such information varies according to the context and the purpose of the application.

3.3. Chapter summary

This chapter described the requirements and proposed an architecture for applications that proactively monitor moving objects. The next chapters detail how to extract and georeference traffic-related facts from tweets, which is the central problem that must be faced by a Dynamic Structured Data Mediator (DSDM), based on tweet feeds, that is part of a truck fleet proactive monitoring application.

4 The monitoring application

As described in section 3.2, a monitoring application must comply with three basic requirements to be proactive: (1) Model the behavior of moving objects and monitor the current state of objects; (2) Model the environment where objects move and monitor the current state of the environment; (3) Detect environment changes that may affect the future behavior of the moving objects and adjust their behavior to the changes. Requirements 1 and 2 are usually fulfilled by countless systems for monitoring moving objects available in the industry. The monitoring application used as motivation in this work also complies with requirements 1 and 2. This chapter aims to briefly describe how these common requirements are implemented in the current application that monitors a fleet of trucks. From the next chapter onward, the focus will be on how to provide proactive monitoring in the application.

4.1. Outline of monitoring application

Consider an application to monitor a fleet of delivery trucks, abstractly defined as follows.

Each truck is modeled as a *moving object M* and each trip is described as a *workflow* W_M that defines the customers to be serviced in the trip and the routes to be followed. Each *step p* of W_M either represents *delivering* merchandize at a customer C_p located at place L_p, or *moving* from a place O_p, called the *origin* of *p*, to a place D_p, called the *destination* of *p*, through a *route* R_p.

For each moving object *M*, the system receives a data stream containing the date, time, geographic position and speed, transmitted by the vehicle at rates that vary from 30 seconds to 5 minutes. The system transforms this raw data into meaningful events with the help of a geospatial database storing the road network and the location of the customers, as well as of other points-of-interest.

An event *E* consists of the occurrence of some fact in a single point of time or during a time interval. Events may be related to moving objects or they may affect their behavior. In the current monitoring application, events are related to moving objects but they are not environment sensitive yet. The environment sensitive behavior is provided by a prototype, explained in details on chapters 5 and 6.

Typically, an event *E* describes how *M* moves. However, *E* may indicate that the execution of W_M has stopped for an unanticipated reason, including an emergency situation. If this is the case, the system triggers an appropriate *emergency workflow* W_E either to allow the execution of W_M to be resumed, if possible, or to propose alternative actions with minimum side effects.

The application monitors several trucks, sharing the same underlying road network and the same emergency workflows. A centralized application is desired to integrate the monitoring of the individual trucks, as well as of the events that affect the road network where the trucks move.

Consider now the problem of improving the truck monitoring application to become proactive and sensitive to the environment.

Briefly, the first change in the application design is to use the truck delivery workflows to infer their future behavior. The second change is to detect anomalies in the conditions of the roads where the trucks are expected to drive in the next steps of their trips (defined by their workflows). As an example, the system may detect that a vehicle, carrying a flammable load, is driving along a road with wet floor ahead. The system may then issue an alert to the driver to proceed more carefully, or to wait in a safe place nearby, or even to take an alternate route.

The application may detect anomalies in the road conditions by analyzing real-time data retrieved from dynamic data sources on the Web, such as Tweets feeds.

Finally, similar scenarios related to other classes of moving vehicles, such as planes and ships may be described. Workflows in this case will be abstractions for flight or sailing plans.

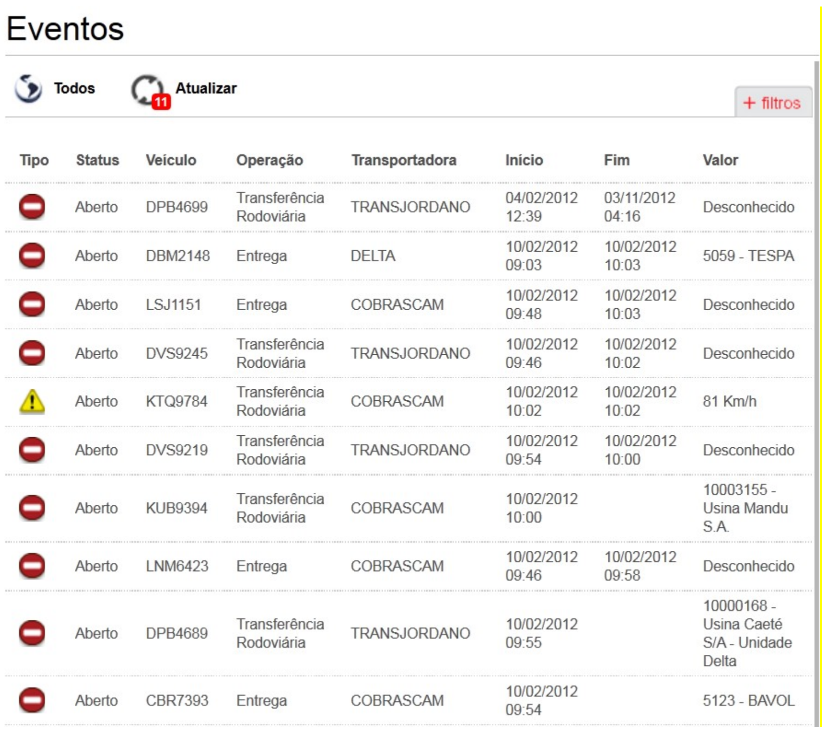
4.2. Moving Objects Monitor

4.2.1. Overview of the Moving Object Monitor

The application currently implements a *Moving Objects Monitor (MOM)* to process the data streams sent by the moving objects to detect certain types of events, that is, to extract trajectory semantics. Figure 3 (in Portuguese) illustrates two types of events that the *MOM* detects: *Stops* and *Speed Limit Exceeded*. For

Stops, column *Value* (*Valor*) represents the location where the object (column *Veículo*) stopped; the location may be an application place (e.g. client or central distribution name) or receive the *unknown* value. For *Speed Limit Exceeded*, column *Value* represents the measured speed value that has inflicted the limit.

The next subsections describe in detail how the data streams are processed.


Eventos

Tipo	Status	Veiculo	Operação	Transportadora	Inicio	Fim	Valor
⛔	Aberto	DPB4699	Transferência Rodoviária	TRANSJORDANO	04/02/2012 12:39	03/11/2012 04:16	Desconhecido
⛔	Aberto	DBM2148	Entrega	DELTA	10/02/2012 09:03	10/02/2012 10:03	5059 - TESPA
⛔	Aberto	LSJ1151	Entrega	COBRASCAM	10/02/2012 09:48	10/02/2012 10:03	Desconhecido
⛔	Aberto	DVS9245	Transferência Rodoviária	TRANSJORDANO	10/02/2012 09:46	10/02/2012 10:02	Desconhecido
⚠	Aberto	KTQ9784	Transferência Rodoviária	COBRASCAM	10/02/2012 10:02	10/02/2012 10:02	81 Km/h
⛔	Aberto	DVS9219	Transferência Rodoviária	TRANSJORDANO	10/02/2012 09:54	10/02/2012 10:00	Desconhecido
⛔	Aberto	KUB9394	Transferência Rodoviária	COBRASCAM	10/02/2012 10:00		10003155 - Usina Mandu S.A.
⛔	Aberto	LNM6423	Entrega	COBRASCAM	10/02/2012 09:46	10/02/2012 09:58	Desconhecido
⛔	Aberto	DPB4689	Transferência Rodoviária	TRANSJORDANO	10/02/2012 09:55		10000168 - Usina Caeté S/A - Unidade Delta
⛔	Aberto	CBR7393	Entrega	COBRASCAM	10/02/2012 09:54		5123 - BAVOL

Figure 3 An example of computation of movement semantics for stops and exceeded speeds

4.2.2. **Analyzing the moving object behavior**

In the example application, the actions of the moving objects are detected and inferred by analyzing the sequences of signals sent by the monitored objects. By processing the signals it is possible to detect and infer some behaviors. The application used as example contains:

1. **Excessive speed** – detected when a signal received has speed higher than 80 km/h.

2. **Excessive driving time** – detected when a vehicle has driven for over four hours without stopping for at least 30 minutes. Each time the server

receives a signal stream from a vehicle, the system checks the time of its last stop with a minimum set duration to identify this type of event.

3. **Stop at client** – inferred from a stop close to a client. Each time a stop is detected, a spatial query is made, using the coordinates of the stop as the center of a circle with radius of 100 meters. The client located closer to this center is associated to the stop.

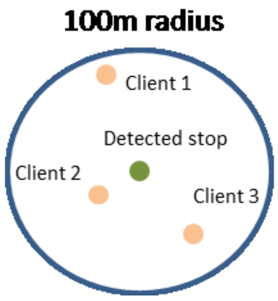


Figure 4 An example of stop at client, in this case the Client 2 is associated to the stop

4. **Forbidden area entered** – detected when the received signals are inside a restricted area registered in the system. These areas represent locations that do not allow truck circulation, such as bridges, residential areas, or expressways.

4.2.3.**Analyzing the moving object status**

Vehicle status detection includes two sets of information: (i) it indicates whether the vehicle is moving or not; (ii) it indicates whether the received signal sequences are recent in relation to time.

4.2.3.1.Movement status

The **stopped** vehicle status is inferred by analyzing the temporal space from the set of signals of the monitored object. The default temporal tolerance used is based on the last 5 minutes. The default spatial tolerance is based on a 50-meter radius and speed above zero. The stop is detected when the signals in a range of 5 minutes do not exit the 50-meter radius from the first signal, and none of these signals has speed higher than zero. The minimum 50-meter tolerance is used to prevent GPS inaccuracy issues. The minimum time of 5 minutes was defined to minimize the detection of false stops (i.e. traffic jams). The spatial and temporal parameters can vary according the scenario.

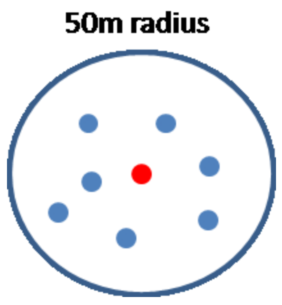


Figure 5 An example of moving object stop detection with the radius of 50 meters around the red point, the blue points are other GPS signals

The **moving** vehicle status is detected when the signals received by the server have speed above 0 km/h.

4.2.3.2.Signals stream up-to-date status

The **no up-to-date signal** vehicle status is detected by the system when no signal is received within 30 minutes in relation to the current time, indicating that the vehicle is not under reliable monitoring. Usually this occurs because the vehicle is moving in an area not covered by the data transmission mechanism, which generally is done via HTTP. Another possible cause is related to problems at the server reception or electric issues in the vehicle or the signal transmitting device.

The **up-to-date signal** status is detected by the system when a signal was received within 30 minutes from the current time, indicating that the vehicle monitoring is reliable.

4.3.
Dynamic Structured Data Mediator

The prototype implementation of the *Dynamic Structured Data Mediator* (*DSDM*) uses Twitter as the main dynamic structured data source. It considers tweets from a predefined list of institutions, assessed as trustworthy sources. Originally, this component was not part of the monitoring application. Hence,it is detailed in this dissertation to cover one of the requirements to make the application proactive.

As illustrated in Figure 6, the *DSDM* receives raw data containing text body, source, user, location (when available), number of re-tweets, hashtags and time stamp. It then filters tweets according to their creation date and keeps only the most recent ones. At the classification step, the *DSDM* selects only the text body and the source. It classifies tweets according to the occurrence of relevant facts in the text body (e.g. car crashes, floods and road blocks). After filtering the relevant tweets, the *DSDM* extracts the spatial reference for the reported fact, with the help of a street gazetteer stored in the *SSDM*. During the experiments, was observed that the locations extracted from the users' tweets are not useful information because they may not represent the actual location of the fact reported. Hence, the *DSDM* extracts addresses from the tweets and uses the Google Maps Geocoding Service as a SSD source to transform addresses into geo-referenced locations. Finally, the *DSDM* transforms the extracted data into a predefined structure before sending the data to the *PCM*.

Chapters 5 and 6 discuss in detail how tweets are processed.

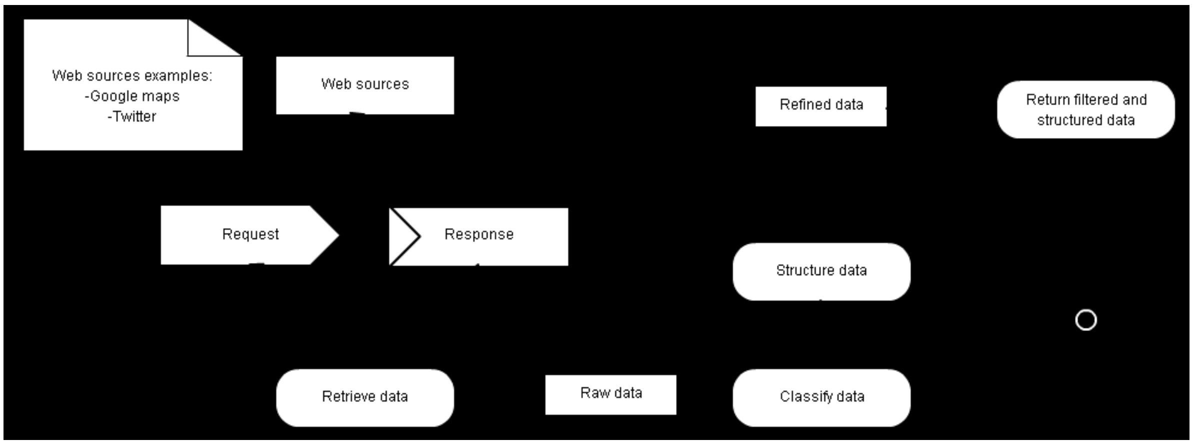


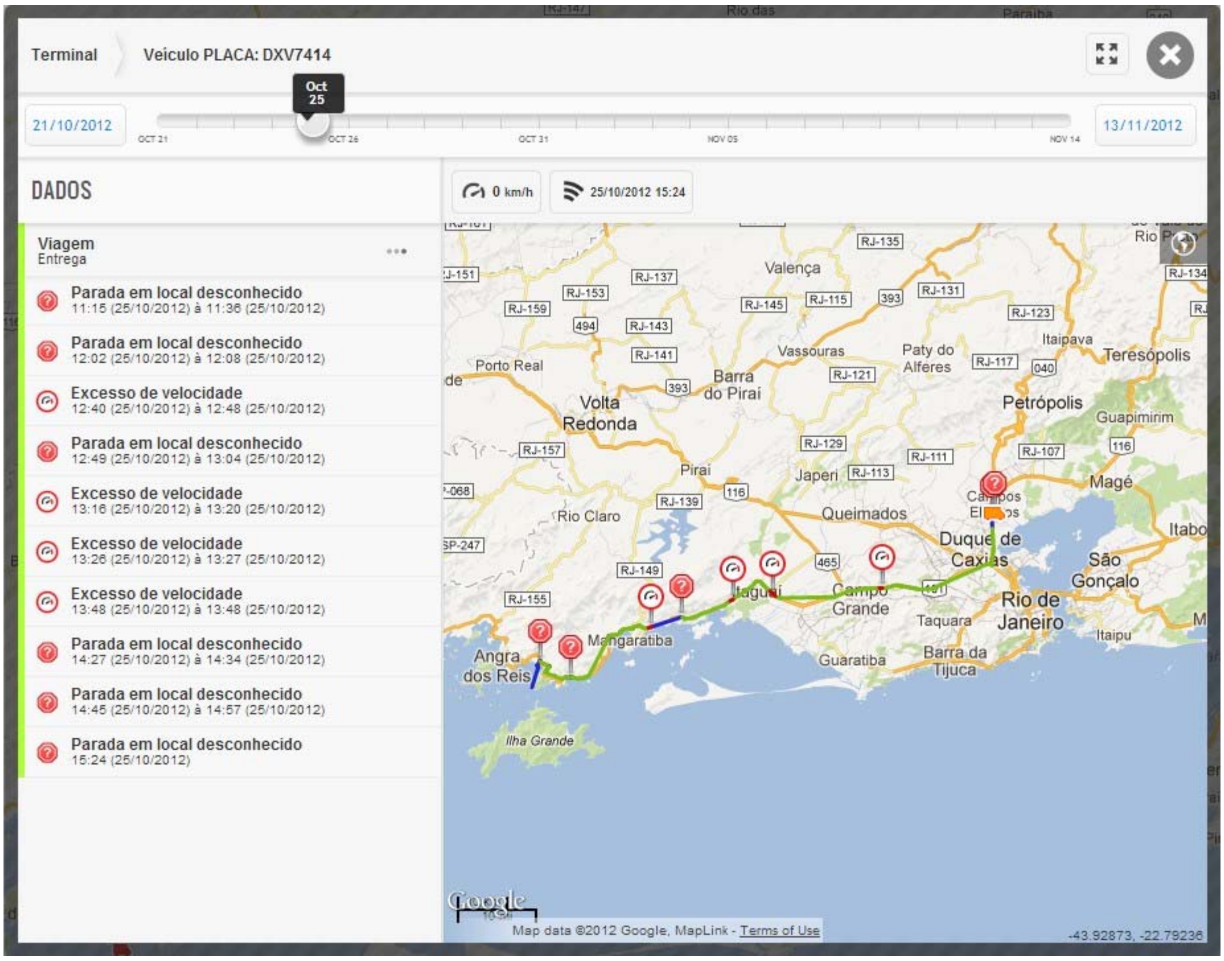
Figure 6 Data flow of the *DSDM*

4.4. Proactive Central Monitor

The implementation of the *Proactive Central Monitor (PCM)* processes facts and events it receives from the *DSDM* and the *MOM*. Currently, the major events the PCM generates are (see Figure 7):

1. **Route deviation** – is detected when the vehicle is over 100 meters away from the programmed route.

2. **Programmed delivery** – is inferred from a **stop at client** event, provided the client is part of the planned route. At each inferred client stop, the system checks whether the client is part of the vehicle's programmed route.



DADOS

Viagem
Entrega

Parada em local desconhecido
11:15 (25/10/2012) à 11:36 (25/10/2012)

Parada em local desconhecido
12:02 (25/10/2012) à 12:08 (25/10/2012)

Excesso de velocidade
12:40 (25/10/2012) à 12:48 (25/10/2012)

Parada em local desconhecido
12:49 (25/10/2012) à 13:04 (25/10/2012)

Excesso de velocidade
13:16 (25/10/2012) à 13:20 (25/10/2012)

Excesso de velocidade
13:26 (25/10/2012) à 13:27 (25/10/2012)

Excesso de velocidade
13:48 (25/10/2012) à 13:48 (25/10/2012)

Parada em local desconhecido
14:27 (25/10/2012) à 14:34 (25/10/2012)

Parada em local desconhecido
14:45 (25/10/2012) à 14:57 (25/10/2012)

Parada em local desconhecido
15:24 (25/10/2012)

Figure 7 Vehicle behavior and environment usage, with some generated events on road network.

Future implementations of the *PCM*, using data provided by *DSD*, will also:

3. **Route replanning.** The PCM will verify if routes are affected by a fact that the DSDM has already sent. If this is the case, the PCM will warn the (human) controller or the driver, or both, that future steps planned for a truck may have to be changed or aborted. The route planning component will be invoked to create routes that consider a list of traffic restrictions.

4. **Incident monitoring.** The PCM will receive events from the MOM that represent incidents involving a truck (e.g. a mechanical problem with M). It then invokes workflows, stored in the TPL, to mitigate the incident and eventual damages to the environment (e.g. to clean up an oil spill).

4.5. Chapter summary

This chapter showed how requirements 1 and 2 from section 3.2 are implemented in the example application. Change detection in the environment where the vehicles are located will be detailed in chapters 5 and 6.

5 Text structuring

Chapter 2 showed some applications using Twitter messages, evidencing the potential value of social networking unstructured databases. This chapter presents an application that reads tweets from special Twitter users which post traffic related messages and structures these tweets.

The tweet structuring process aims to identify facts that might influence the future behavior of moving objects and discover where they occur. This chapter and the next are the main contribution of this work.

5.1. Solution overview

This dissertation focuses on Twitter as case study due to the large number of users, its great data generation capacity, and its consolidated position in the industry. Specifically, it describes how to retrieve tweets from a given list of profiles and interprets the sentences retrieved, in order to learn about traffic conditions and the occurrence of certain events of interest. This would allow for low-cost, collaborative monitoring.

The profiles selected are both governmental and private (companies) which publish traffic-related information. The example shown in Fig. 8 represents the tweet: "*Av Maracanã, sentido Centro, tem trânsito intenso na altura da R São Rafael, por causa de obra na via. #zonanorte*" (translated as "Maracanã Ave., direction Center, with heavy traffic near São Rafael St. due to road construction. #northarea"), published in 03/26/2012 by a governmental source (@OperacoesRio). The illustration highlights parts of the sentence that are relevant to monitor traffic conditions.

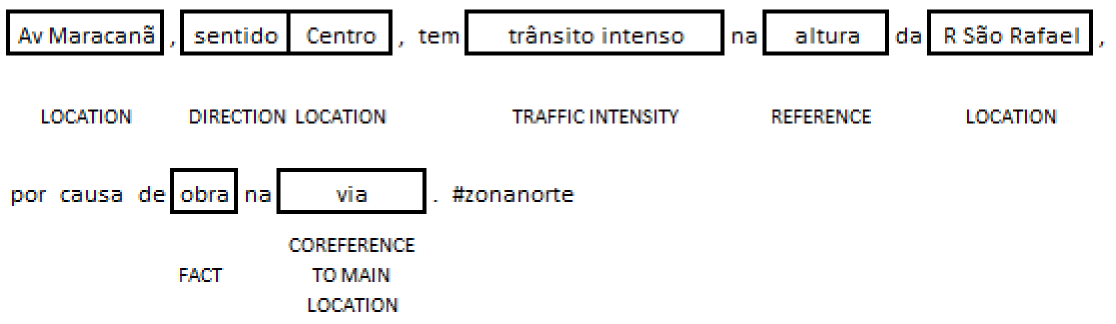

Av Maracanã , sentido Centro , tem trânsito intenso na altura da R São Rafael ,
LOCATION
DIRECTION
LOCATION
TRAFFIC INTENSITY
REFERENCE
LOCATION
por causa de obra na via . #zonanorte
FACT
COREFERENCE TO MAIN LOCATION

Figure 8 A real traffic-related tweet (in Portuguese) with its entities

Using a conventional system based on pre-programmed rules and conditions to solve this problem is costly. For instance, Endarnoto et al. (2011) propose a template-based solution to extract events of interest and their respective locations from messages with a common format and limited grammatical variation. However, the scenario addressed in this work involves sentences with high grammatical variation. In order to deal with grammatical variation, produce acceptable results and minimize application maintenance and update costs, this work proposes a solution based on Artificial Intelligence, using natural language processing and machine learning techniques.

The problem is decomposed into two tasks: (i) identifying and annotating the relevant entities in the text; and (ii) interpreting the text and extracting relevant facts using the entities identified. The first task is related to the named entity recognition (NER) problem, and the second to the information extraction (IE) problem.

The NER problem is widespread in the literature [Nadeau et al. 2007], including in the context of using Twitter as data source. Ritter et al. (2011) propose an entity recognition approach using Twitter, calling attention to idioms used in the Internet. Jung (2012) also uses Twitter as case study.

The IE task consists in identifying and extracting certain information or details from a document. IE is important, for instance, in the context of question and answer systems, where the information requested by the user must be recognized in a data or text base, and in automatic summary systems, in cases where the summary to be produced focuses on events or concepts described in the text [Mani and Bloedorn 1998].

5.2. Structuring Twitter data

5.2.1. Named Entities

This work uses the following list of entities. The type of entity is followed by the description and its objectives.

Location (LOC): Indicates a location that can be georeferenced. It can be a country, a state, a neighborhood, a point of reference, or any other physical location. Also includes buildings and facilities such as airports, shopping centers, etc. A street name can be, for instance, "Rua Marquês de São Vicente" (Marquês de São Vicente street), where all of the words are used as the location. Points of reference that indicate a point in space also must be annotated as a location. This situation includes information about a specific kilometer on a highway, a road exit, or any known location, such as "km 135", "exit 5" or "Santos Dumont Square".

Point of reference (REF_N): Represents the proximity to a more specific location. This entity connects a primary location to a secondary one in order to increase the precision of the location. Words indicating proximity and enhancing the precision of the principal location are used as reference.

Lane direction (DIR_N): Indicates the flow direction. A piece of information about an accident on a two-way expressway is not complete. Words representing this entity indicate the traffic flow direction affected by an event of by traffic intensity. Based on the flow direction, it is possible to know which road lane is affected, for instance. Just like the REF_N entity, DIR_N connects a principal location to a secondary one, where the secondary one must indicate the destination of the affected lane.

Both lanes (DIR_NS): Indicates an event affecting both directions. This entity must be used to represent words that express a fact or traffic intensity in both ways (both lanes) of a road.

Co-reference (COREF_LOC): Indicates a co-reference to the principal location. In some cases, one location can be the target of several events, such as a fact that causes heavy traffic. In this context, the purpose of the entity is to refer to the principal location of the primary event.

Restrictive location (ABSL_N): Indicates a location whose purpose is to restrict a geographic area related to a given primary location. This entity connects a principal location to a secondary one, indicating a restrictive location for the primary one. Such entities usually indicate the neighborhood or city of a given street of location.

Traffic intensity (TR_INT): Represents the intensity of the traffic flow. The set of words annotated by this entity must relate to the traffic conditions of a given location. It is sub-classified into TI_1, TI_2 and TI_3, meaning good, heavy, and slow, respectively.

Good traffic (TI_1): Represents traffic in its best condition, without any retentions or jams.

Heavy traffic (TI_2): Represents intense traffic, with retentions and minor points of traffic jams.

Slow traffic (TI_3): Represents traffic in its worst condition, with several jams and very slow.

Fact (FACT): Indicates general events that have some kind of impact on the traffic. These are usually related to road blocks, accidents, collisions, pedestrian incidents or emergencies in general. The words associated to this entity should be enough to express a fact that is relevant to traffic and changes the normal status of the flow. It is sub-classified into ACC, EMER, and HINDER, meaning accident, emergency, and hindering fact, respectively.

Accident (ACC): Indicates the occurrence of accidents, pedestrian incidents and related events.

Emergency (EMER): Indicates the existence of relevant facts that compromise the security of the drivers, such as heavy rain, landslides, roads with open manholes, lack of lighting, defective traffic lights, and so forth.

Hindering fact (HINDER): Indicates the occurrence of facts that hinder the flow on a given road. These include construction work, partial or total road blocks, protests, etc.

End of a fact (OPEN): Indicates the end of an event. This could be the removal of a collided vehicle, the opening of a road block, the removal of a fallen tree after heavy rain, etc. The words associated to this entity are usually verbs in the past tense.

Other (O): Entities which are not relevant to the problem. Must be used when a given word does not fulfill any of the previous requirements.

The **Traffic intensity** and **Fact** entities may also be sub-classified in order to define the intensity level of traffic or determine the type of fact by means of a particular set of words or a more generic mechanism that uses Web resources (e.g. WordNet).

A common problem when recognizing named entities is in identifying where an entity begins and ends, since an entity can be described by one or more words. To use a real example, consider the set of words "**Praça Santos** Dumont" (Santos Dumont Square). The system will identify the words "Praça Santos" as the **LOC** entity, but would fail in identifying the end by ignoring "Dumont", classifying it as **Other**. In a case like this, "Praça Santos" is classified as an incorrect positive prediction (false positive) and classify "Dumont" as an incorrect negative prediction (false negative). This way, a single entity generates two incorrect entries simply by failing to identify their boundaries.

.

5.2.2. **Information Extraction**

This work models the expected result of the IE problem, also known as relationship extraction, as a dependency tree. Each tree node corresponds to a Relevant Entity, which represents the set of all entities used in this work, except for **Other**. The edges between parent and child in the tree represent a dependency relationship between the entities. Structuring news as trees makes them easy to use in different applications (e.g. for georeferencing and associating locations and facts). Fig. 9 illustrates all possible relationship restrictions among the entities created in this work. These restrictions were developed considering the characteristics of each entity. The ROOT type corresponds to the tree root and does not represent an entity.

.

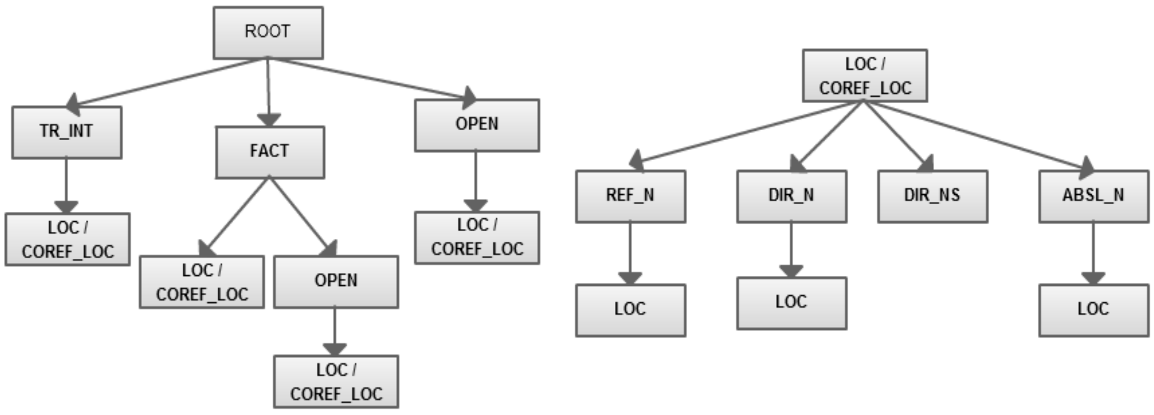


Figure 9 Relationship restrictions among relevant entities

Applying the tree rules illustrated in Fig. 9, a set of real sentences, which show the relationships among entities, is presented below. Words to the left represent the parent entity, followed by its relation nature, and finally the child entity.

Example 1:

Obra no Viaduto de os Pracinhas, sentido Centro. Transito intenso no local.

FACT LOC DIR_N LOC TR_INT COREF_LOC

(Construction at Pracinhas Overpass, direction Center. Intense traffic in the area.)

Construction [*where*] Pracinhas Overpass

Pracinhas Overpass [*direction*] direction

Direction [*which*] Center

Intense traffic [*where*] area

Example 2:

Lentidão em a Rua Jardim Botânico, sentido Gávea.

TR_INT LOC DIR_N LOC

(Slow traffic at Jardim Botânico Street, direction Gávea.)

Slow traffic [*where*] Jardim Botânico Street

Jardim Botânico Street [*direction*] direction

Direction [*which*] Gávea

Example 3:

Trânsito intenso na Rua Jose de o Patrocínio, altura de o No382, sentido Tijuca.

TR_INT LOC REF_N LOC DIR_N LOC

(Intense traffic at José do Patrocínio Street, around number 382, direction Tijuca.)

Intense traffic [*where*] José do Patrocínio Street

José do Patrocínio Street [*where*] #Grajaú

José do Patrocínio Street [*near*] around

Around [*where*] number 382

José do Patrocínio Street [*direction*] direction

Direction [*which*] Tijuca.

5.3. Data structuring process

Algorithms in the field of machine learning typically do not receive text as input. To use text with these algorithms, the words need to be converted into a structure called “bag of words” [Lewis 1998] [Maron 1961], where a vector representation for the words is created.

In order to improve the output quality of the proposed machine learning tasks, some attribute generators that seek to extract additional information about each input object was developed. In the literature, such attributes are known as features.

A token is defined as a string of alphanumeric characters or a single punctuation mark.

5.3.1. Attributes – Entity recognition

In this task, each token has a set of characteristics which are used by the machine learning mechanism to define its entity. Consider $X_i = \{X_1, X_2, \ldots, X_n\} \in T$, where T is the list of tokens related to a piece of news. The possible variations of X are:

Token (W_i) – Indicates the content of token i.

Simple Token (SMW_i) – Indicates the simplified content of token i (in lower case, without any special characters and punctuation).

Simplified Token (SW_i) – Indicates the simplified content of token i (in lower case, without any special characters, and removing letters, punctuation or numbers of size 1).

Part-of-Speech (POS_i) – Indicates the part of speech of token i.

Stemmed Word (STW_i) – Indicates the root of the word present in token i.

Ex. 1: The word *blocked* is turned into *block*.

FactLike – Indicates whether SMW_i is listed in a dictionary of reserved words referring to the fact.

The list of attribute generators used to recognize entities is the following:

CurrT(X) – Current token: X_i.

PrevT(X, N) – Represents the N tokens before the current index, where each token represents an attribute: $X_{i-1}, \ldots, X_{i-N}$.

NextT(X, N) – Represents the N tokens after the current index, where each token represents an attribute: $X_{i+1}, \ldots, X_{i+N}$.

CurrWSC – Indicates whether W_i begins in upper case and does not have any other lower-case letter.

Ex.1: Avenue – true

Ex.2: AveNue – false.

LocType – Indicates whether the lower-case W_i indicates a designation: $X_i \in$ {av, ave, avenue, avenues, hwy, highway, r, rd, road, roads}.

Below is an example of a set of attributes of a token from the Entity Recognition dataset.

Sentence: Rua São Clemente, em Botafogo, com trânsito bom na altura do Consulado Português, na #zonasul. (Translation: São Clemente Street, in Botafogo, has good traffic near the Portuguese Consulate, in #southarea.)

Considering i = 3:

CurrT(W_i): Clemente	**CurrT(SW_i)**: clemente	**CurrT(POS_i)**: NPROP
PrevT(W_i, 2): Rua	**PrevT(POS_i, 2)**: NPROP	
PrevT(W_i, 1): São	**PrevT(POS_i, 1)**: NPROP	
NextT(W_i, 1): ,	**NextT(POS_i, 1)**: ,	
NextT(W_i, 2): em	**NextT(POS_i, 2)**: PREP	

LocType(i): NO_LOCAL_TYPE

CurrWSC(i): YES_SPECIAL_CAPITAL

5.3.2. Attributes – Information extraction

In this task, each piece of news (one tweet) is transformed into a directed graph with N nodes, where each node is represented by a relevant entity. The possible number of edges is $|N|^2$-$|N|$, and each edge represents a pair of relevant entities, where A -> B indicates that A is B's parent. Each relationship between two entities has a set of attributes, which is used by machine learning algorithms to learn about the nature of these relationships. Consider $Y_i = \{Y_1, Y_2, \ldots, Y_n\} \in$ ENT, where ENT is the list of entities in the piece of news. The variations of Y are:

Word (W) – Indicates the words of Y_i.

Simplified Word (SW) – Indicates the simplified words of Y_i.
Ruler Entity (RE) – Indicates the named entity of Y_i; if it is not a Relevant Entity, POS is used.
Named Entity (NE) – Indicates the named entity of Y_i; is ignored if it is not a RE.
Punctuation (PUNCT) – Indicates whether there is punctuation.

The list of attribute generators used for information extraction is the following:

PossRel – Indicates whether the pair of entities may have a dependency relationship.
ConcT (Y) – Represents the concatenated words of Y_i -> Y_j.
BetT (Y) – Represents all tokens in the range Y_i , Y_j.
NearT (Y, N) – Represents the tokens in range N, where: Y_{i-1}, …, Y_{i-N};Y_{j-1}, …, Y_{j-N} ;Y_{i+1}, …, Y_{i+N}; Y_{j+1}, …, Y_{j+N} . It counts as a single attribute, separated by character "_".
AbsLocPair – Indicates whether entities A -> B are related, where A = {ABSL_N} and B = {LOC}; otherwise, no value is used.
MetaWithLoc – Indicates whether entities A -> B are related, where A = {REF_N ou DIR_N} and B = {LOC}.

Below is an example of a set of attributes referring to a pair of entities, Local and Traffic Intensity, respectively, from the Information Extraction dataset.

Sentence: Rua São Clemente, em Botafogo, com trânsito bom na altura do Consulado Português, na #zonasul. (Translation: São Clemente Street, in Botafogo, has good traffic near the Portuguese Consulate, in #southarea.)

Considering Y_i -> Y_j , where i = 4 and j = 1.

PossRel: PR_YES
Conc (NE): TI_1_LOC
ConcT (SW): transito bom_rua sao clemente
BetT (RE): ABSL_N LOC , PREP
BetT (W): em Botafogo , com
NearT (RE, 3): ABSL_N_LOC_PREP_, PREP_ REF_N_PREP
NearT (SW, 2): com___em_a_ _ _em_

NearT (NE, 2): LOC_ABSL_N_ REF_N_LOC_

AbsLocPair: _ ABSL_N_LOC_

5.4. Corpus

This section discusses the procedures employed to construct the corpus, which is a set of examples manually annotated, used in the experiments described in next section.

As explained in chapter 1, the data for the analyses carried out in this work was gathered from Twitter. For a first version, two data sources were selected, which correspond to two Twitter users, to extract news about traffic and its causes. The selected users were *@operacoesrio* and *@odia24horas*. These data sources publish news only about Rio de Janeiro, most of them (around 90%) traffic related, and use a rather formal style, without many idioms or typically web-related language uses. This ensures the construction of a base with news of a certain kind, so that, in another stage of the project, other data sources which are not necessarily related to traffic can be gathered and interpreted.

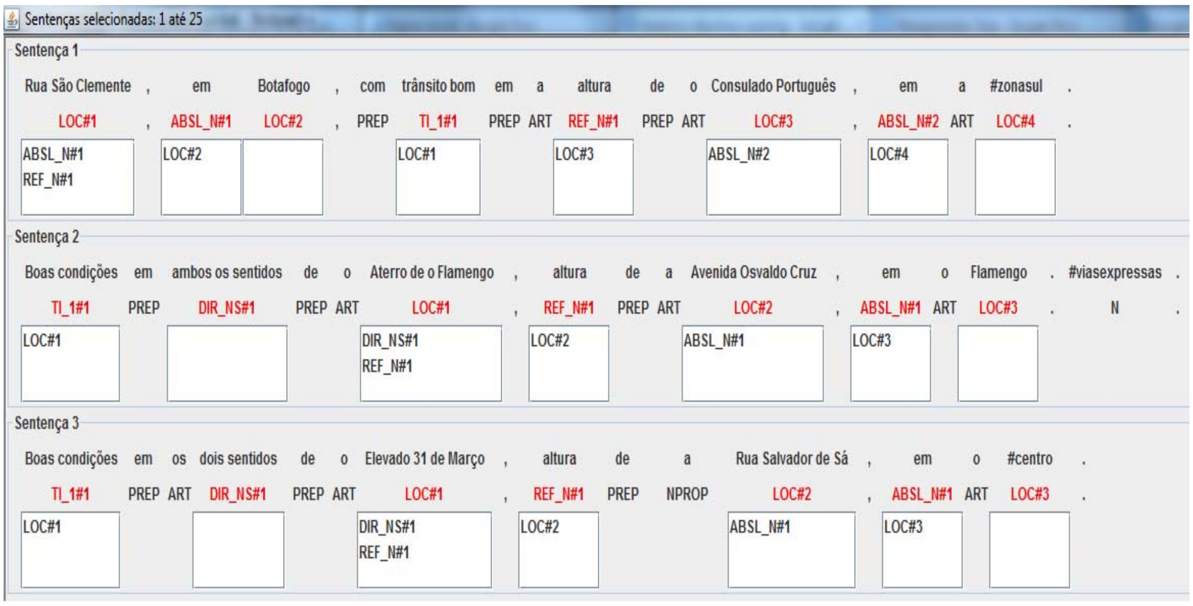
Corpus preparation was divided into the following steps:

(a) extracting the tokens from each sentence;
(b) defining the morphosyntactic characteristic of each token;
(c) associating each token to an entity;
(d) establishing the dependency relationships between entities.

To execute steps (a) and (b), an automatic procedure was developed, using the Webservice called F-EXT [Mota et al. 2010]. It allows gathering all the tokens from a sentence, with their morphosyntactic characteristics. Step (c) was initially done automatically by a program with certain rules and then verified by a person. To reduce the time spent on this procedure, this algorithm was replaced with the entity identification mechanism based on machine learning, using 100 sentences with annotated entities. As a result, the verification time dropped from two hours to one hour to analyze 100 sentences. Step (d) was done manually, using a graphics interface with drag and drop capabilities (illustrated in Fig. 10), to help view the dependency relationships. Steps (c) and (d) were carried out using the rules described in section 5.2, being characterized as gold annotations because they were done by a person.

The current corpus has a total of 475 annotated pieces of news, with an average of 23.5 tokens per piece of news. Table 1 shows the relationships among entities and the tokens present in the corpus. Table 2 shows corpus information from the perspective of the dependency trees. Table 3 represents some tree corpus statistics.

.



Sentenças selecionadas: 1 até 25

Sentença 1

Rua São Clemente , em Botafogo , com trânsito bom em a altura de o Consulado Português , em a #zonasul .

LOC#1 , ABSL_N#1 LOC#2 , PREP TI_1#1 PREP ART REF_N#1 PREP ART LOC#3 , ABSL_N#2 ART LOC#4 .

ABSL_N#1 REF_N#1 | LOC#2 | | LOC#1 | LOC#3 | ABSL_N#2 | LOC#4 |

Sentença 2

Boas condições em ambos os sentidos de o Aterro de o Flamengo , altura de a Avenida Osvaldo Cruz , em o Flamengo . #viasexpressas .

TI_1#1 PREP DIR_NS#1 PREP ART LOC#1 , REF_N#1 PREP ART LOC#2 , ABSL_N#1 ART LOC#3 . N .

LOC#1 | | DIR_NS#1 REF_N#1 | LOC#2 | ABSL_N#1 | LOC#3 |

Sentença 3

Boas condições em os dois sentidos de o Elevado 31 de Março , altura de a Rua Salvador de Sá , em o #centro .

TI_1#1 PREP ART DIR_NS#1 PREP ART LOC#1 , REF_N#1 PREP NPROP LOC#2 , ABSL_N#1 ART LOC#3 .

LOC#1 | | DIR_NS#1 REF_N#1 | LOC#2 | ABSL_N#1 | LOC#3 |

Figure 10 Graphics interface of the entity dependency annotator

Entity	By token	Complete entity
ABSL_N	105	104
COREF_LOC	107	107
DIR_N	316	311
DIR_NS	114	46
FACT	492	243
LOC	2898	1461
O	6045	6045
OPEN	71	64
REF_N	258	255
TR_INT	737	393
Total	11158	9039

Tabela 1 Figures on the relationships between entities and tokens.

Entity	By token	Complete entity
TI_1	166	68
TI_2	387	221
TI_3	184	104
ACC	83	75
EMER	111	42
HINDER	298	129

Table 2 Sub-classification of entities FACT and TR_INT.

Edges	# Possible edges	In relation to total	In relation to reduced total
Combinations	24652	100.00%	297.00%
Reduced combinations	8325	33.60%	100.00%
Connected	2992	12.10%	36.00%

Table 3 Tree statistics.

5.5. Quality measures

To measure the results, the ten-fold cross-validation technique was used. The corpus was divided into ten equal parts and tested ten times, where at each test one part is used in the testing set and the other nine are used in the training set. After the ten tests are executed, the result of the quality measurements is an average of the result of all ten executions. The quality measures used were accuracy (Acc), recall (Recall), precision (Prec), f-measure (F-1), and standard deviation of f-measure (D. P. F-1).

The precision and recall metrics are most commonly used in binary classifications. The binary classification task is often more similar to filtering than to clustering, because few positive cases are identified among a large amount of negative cases. Considering the collection task, this situation can be clearly observed. In long texts, the objective is collecting only the parts that can be used as learning objects. Precision and recall correctly assess this task, and, given a category c_i, precision and recall associated to this category are defined as:

$$\text{Precision} = \frac{tp}{tp + fp} \qquad \text{Recall} = \frac{tp}{tp + fn}$$

Where:

		Gold annotation	
		True	**False**
Test outcome	**Positive**	True positive (tp)	False positive (fp)
	Negative	False negative (fn)	True negative (tn)

These two global measures usually yield considerably different results due to the distribution of examples among the classes. In this case, each entity defined in chapter 5.2 corresponds to one class. If the distribution is too disproportionate, which happens very often, the measures by class will be as important as the global measures.

For a classifier to be considered good, it is not enough that only one of these measures is high. For this reason, a third measure is also computed, which makes a joint assessment of the measures, as follows:

$$F_\beta = \frac{(\beta^2 + 1)\, precision \times recall}{\beta^2\, precision + recall}$$

The assignment β = 1 is used, attributing the same importance to both measures, and thus computing the metric called f- measure (F_1).

The Accuracy measure on its own is not a good performance measurement, because higher values could be achieved by always classifying texts into the negative class.

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

Weka version 3.6.5 was used for entity recognition. The algorithm that generated the best result among several tested was the SMO implementation [Platt 1998], of the SVM Family. To extract the dependency tree, an implementation (available at https://github.com/eraldoluis/Large-Margin-Structured-Perceptron) was employed which:

(i) uses the large margin structured perceptron to compute a weight for each edge, forming a directed graph with weighted edges;

(ii) finds the maximum generating tree in the directed graph [Chu and Liu 1965][Edmonds 1967].

All tests were performed in a computer with a 2.13 GHz Core i3 processor, with 4Gb of RAM. Tables 4 and 5 represent the best results obtained using the attributes described in section 4.3.

Execution time: 952.3 seconds.

Features used: CurrT(W), CurrT(SW), CurrT(STW) PrevT(W, 2), NextT(W, 2), PrevT(STW, 1), NextT(STW, 1), PrevT(POS, 2), CurrT(POS), NextT(POS, 2), LocType, CurrWSC, CurrT(FactLike), PrevT(FactLike, 1), NextT(FactLike, 1), PrevT(SMW, 1), NextT(SMW, 1).

By Token						Set of tokens (by boundaries)				
Entidades	Acc	Prec	Recall	F1	D.P. F-1	Acc	Prec	Recall	F1	St. Dev. F-1
ABSL_N	99.85	81.74	87.49	84.15	4.56	97.21	100.00	97.21	98.54	2.20
COREF_LOC	99.95	97.62	97.56	97.52	2.77	97.56	100.00	97.56	98.73	1.94
DIR_N	99.57	94.36	90.87	92.36	8.16	89.91	99.72	90.03	94.24	7.13
DIR_NS	99.99	100	98.75	99.33	2.00	92.50	93.33	95.00	94.00	18.00
FACT	98.74	88.86	82.24	85.10	5.80	45.51	70.74	55.01	60.62	16.84
LOC	98.26	96.41	96.82	96.59	1.14	86.69	94.43	91.28	92.79	2.90
O	95.76	95.49	96.79	96.13	0.92	97.48	100.00	97.48	98.72	0.61
OPEN	99.88	98.57	87.12	92.01	8.40	78.29	91.75	82.67	85.60	16.97
REF_N	99.80	96.95	92.92	94.69	4.34	92.01	100.00	92.01	95.62	4.79
TR_INT	99.55	97.11	96.35	96.70	2.71	84.13	93.38	89.66	91.09	5.82

Table 4 Best results for entity recognition

By Token						Set of tokens (by boundaries)				
Entidades	Acc	Prec	Recall	F1	D.P. F-1	Acc	Prec	Recall	F1	St. Dev. F-1
TI_1	99.80	96.38	90.38	92.77	4.29	70.11	87.78	74.90	79.76	18.70
TI_2	99.77	96.63	96.97	96.70	2.23	90.11	96.87	92.79	94.66	3.79
TR_3	99.81	98.06	94.13	95.72	4.82	87.57	99.33	87.87	92.06	12.68
ACC	99.83	94.64	88.60	89.75	11.55	86.57	100	86.57	90.89	15.79
EMER	9.01	33.11	12.15	16.06	5.63	9.01	33.11	12.15	16.06	5.63
HINDER	35.51	60.00	43.48	48.50	20.13	35.51	60.00	43.48	48.50	20.13

Table 5 Results of the sub-classifications of entities FACT and TR_INT

Execution time: 10.04 segundos.

Features used: PossRel, ConcT(NE), ConcT(W), BetT(RE), BetT(W), BetT(NE), NearT(RE,3), NearT(W,2), NearT(NE,2), AbsLocPair, MetaWithLoc, BetT(PUNCT)

Edge reduction	Accuracy by edge	Accuracy by example	Standard deviation of the accuracy by example
NÃO	92.68	69.38	9.45
SIM	93.16	73.37	8.41

Table 6 Best results for tree prediction

5.6. Chapter summary

This chapter described a solution to transform unstructured traffic related text messages, from Twitter, into valuable structured data. The solution is based on machine learning techniques and it has two phases: (a) uses a NER (Named Entity Recognizer) to discover relevant entities in text message; (b) creates a dependency relationship over discovered entities, to produce structured data. After these two phases, the produced data can be used by an agent or an application to discover what and where the facts occurred.

6
Using structured traffic message

From now, we address the problem of inferring the location of facts that affect road conditions by analyzing real-time data retrieved from dynamic data sources on the Web. In general, the location of such facts is useful for real-time applications that monitor moving objects and that support decision making. For example, car crashes and road blocks are relevant to such applications because they affect the traffic flow by reducing the average speed and imposing changes on the planned route. However, to be useful, the location of such facts must be estimated as accurately as possible. Furthermore, they must be provided as timely as possible, which justifies exploring dynamic data sources on the Web.

The most common way to georeference locations is to use geocoding techniques, which can be defined as a process to estimate the most accurate location for a set of geographic points from locational data such as postal code, street name, building name, neighborhood, etc. As summarized by Goldberg, Wilson and Knoblock (2007), geocoded data that used to cost \$4.50 per 1,000 records as recently as the mid-1980s, quickly moved to \$1.00 by 2003, and can now be done for free, using online services which usually have limitations such as a maximum number of requests per day. Actually, per day, Yahoo! PlaceFinder allows up to 50,000 requests, while Google allows 2,500 requests, Bing allows 15,000 requests, and CloudMade provides this service completely for free.

Information about a location can be imprecise and context-dependent. In a road network, where streets may be long or two-way, just the name of a street may represent low-valued information for certain applications. To improve precision, geocoding commonly includes the number of a building on the street, the highway location, often indicated in kilometers, or the postal code. Another way to reference locations, frequently used in human communication, is to use a proximity attribute, declaring that location *A is near* location *B*, rather than directly using the address of location *A*. Another relevant aspect of location description using natural language is the direction attribute, i.e., the street direction towards a location.

In this section, we outline a model to georeference the location of facts, using geocoding and routing services, from spatial descriptions commonly found in human communication. To validate the model, a prototype application that uses structured traffic-related news in natural language to infer locations is described. This application is part of section 3 architecture [Albuquerque et al. 2012 [2]].

6.1. Naive geocoding usage

To motivate the solution discussion, consider the following scenario. Every day, Twitter text messages (“tweets”) with traffic-related contents are published by institutional or individual users as a collaborative initiative. Institutional tweets, such as those published by CET-RIO, are fairly well-structured and can be used as raw input data in the context of our target application. Each traffic-related tweet contains one or more simple facts (such as traffic intensity) or describes events (such as accidents or road blocks) and their respective location. Simple facts and events are not distinguished here, and refer to both as facts. Retrieving these associations from raw text is not trivial because there is no commonly expected format or template for natural speech. In order to associate the facts to their accurate locations, the traffic-related fact structure is used, as explained in Section 4.

The naive use of location as input to the georeferencing process may produce imprecise results. As an example, consider the text illustrated in Figure 11: “Car accident on street A, located at district K, in the direction of Hospital X, near street B”. Suppose also that street A is a two-way street and is 5 kilometers long.

If the geocoding process outputs only “street A”, then the information will be quite inaccurate: it is not possible to know the exact location of the accident along the 5 kilometers, or in which street direction it occurred. On the other hand, a geocoding service that qualifies “street A” with “near street B” provides valuable information that can be used to narrow the location of the accident, whereas the text fragment “in the direction of Hospital X” indicates which street direction was affected.

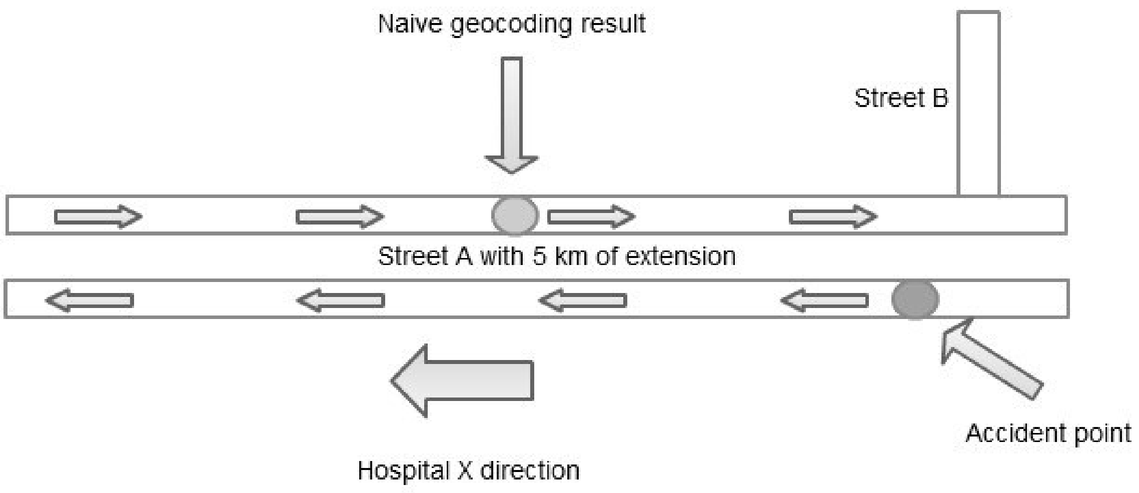


Figure 11 Example of naive geocoding

The use of additional predicates, based on spatial references, also helps improving the description of a location. In the example above, it is easier for a driver to identify *Hospital X* along a street than to check the number of the buildings. Once the hospital location is known, spatial reasoning will provide additional information. Therefore, references like *near*, *intersecting*, and *located at*, although not deterministic, narrow the scope of the location-based analysis.

6.2. Proposed geocoding model

This section presents the proposed geocoding model and how it is used to increase georeferencing precision, relying on geocoding and routing services available on the Internet.

As discussed in the introduction of Chapter 5, additional data is tipically used to improve the precision of a location of interest. The adopted model is summarized in Figure 9, has the following entity sets and relationships:

Entity Sets

Fact	the set of all relevant facts (such as *"slow traffic"* and *"car crash"*)	
Location	the set of all relevant locations	
	Name	a string attribute assigning a name to the location.
	Geometry	a 2D attribute assigning a geometry to the location.
POI	the set of all places-of-interest, a specialization of **Location** (such as *"North Shopping*" and *"West Hospital*")	
Street	the set of all relevant streets, a specialization of **Location** (such as "*Main Street"*)	

Two-way a Boolean attribute which is true when the street is two-way.

Relationships

occurs relates a fact to a single location, where "*F* **occurs** *X*" indicates that *F* is a fact that occurs in a location *X*, in which case we say that X is the *main location of interest* for *F*.

Both a Boolean attribute which is true when *X* is a two-way street and

F affects *X* in both directions.

qualifies relates a street *X* to a location *Y*.

How an attribute with one of the following 3 values:

direction indicates that *Y provides a reference direction* for *X* (such as "*Main Street in the direction of the North Shopping*").

restriction indicates that *X is restricted to* (such as "*Main Street restricted to the South Borough*").

reference indicates that Y provides a reference location for X (such as "Main Street having as a reference the West Hospital")

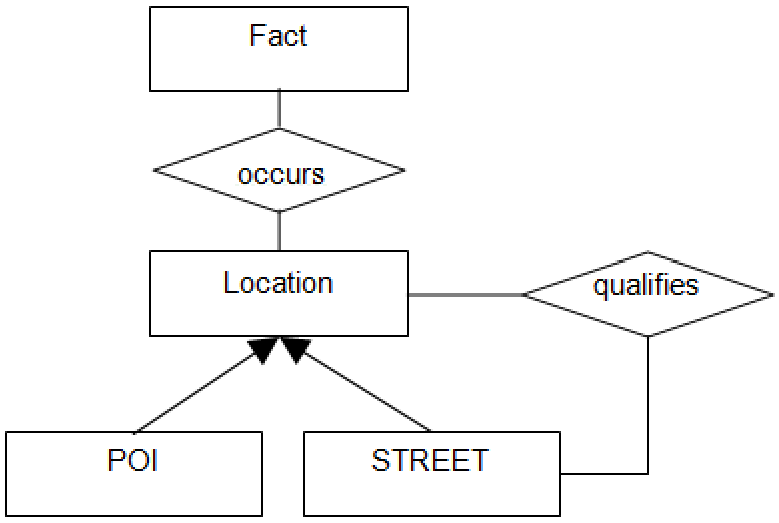


Figure 12 Simplified entity-reationship diagram of the geocoding model

The model usage is as follows:

Let *F* be a fact that occurs on a location *M*, called the *main location of interest*.

Assume that *M* is restricted by a location *A* and that the geometry of *A* is a polygon. Then, *A* must be used to filter *M* in two different ways: (i) by geocoding the boundaries of *A* and using them to filter *M*; or (ii) by appending the location name of *A* to the main location *M*.

Assume that *M* is a two-way street and that *D* provides a reference direction for *M*. Then, a routing service must be called, passing as parameters *M* as the origin and *D* as the destination, to discover a route *r* that goes from *M* to *D*. Then, *r* must be used to simplify the geometry of *M* to just the affected direction.

Assume that *M* is a street and that *R* provides a reference location for *M*. Then, the geometry of *R* must be used to again simplify the geometry of *M*. For example, if the geometry of *R* is a point (i.e. a building), the parts of the geometry of *M* that lie outside a circle of a given diameter whose center is the geometry of *R* may be discarded.

The proposed geocoding model pseudo-code is follows:

```
SmartGeocode(M)
    M.geometry = geocode(M.location)
    for each Di in M.D
        if Di contains location
            Di.geometry = route(M.location, Di.location)
    for each Ri in L.R
        for each RiLocation in Ri.locations rGeom
            = geocode(RiLocation.location)
            Ri.geometry = Ri.geometry.union(rGeom)
    for each MGeom in M.geometry
        for each Di in M.D
            if MGeom.intersects(Di.geometry) or MGeom.touches(Di.geometry)
                auxGeom = auxGeom.union(MGeom)
    if auxGeom is not null
        M.geometry = auxGeom
    for each Ri in M.R
        minimumDistance = MED(M.geometry, Ri.geometry)
        auxRefCoordinate = minimumDistance.E0
        L.referencePoints.add(auxRefCoordinate)
```

Figure 13 illustrates the result of applying this process to the text example described in Section 6.1.

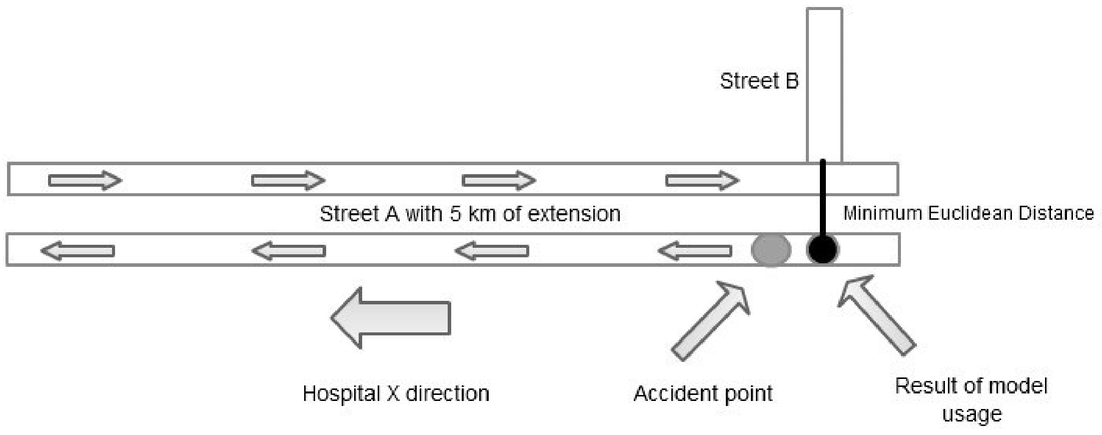


Figure 13 More precise result with proposed model

6.3. Georeferenced data process

The prototype application implements the process outlined in Section 5.1 to georeference the locations of traffic-related tweets.

The JTS Topology Suite (Aquino and Davis, 2003) is used, a Java open-source API that implements many 2D geometry functions. Some of these functions and common geometry types are summarized in Bressan and Zhang (2005), which also propose a benchmark for XML processing in GIS applications.

CloudMade and Google provide the geocoding and routing services. CloudMade offers tools and APIs to develop location-based applications, including geocoding and routing services, using the Open StreetMap (OSM) database. An advantage of using OSM is that this service returns the geometry of roads and buildings (e.g. for a road, it returns a line or multiline and, for a building, it returns either its coordinates or the polygon contour, whichever it is available). The geocoding and routing services provided by Google act as a backup resource: they are used when CloudMade cannot find a valid geometry for the desired geocode location or route. Google's geocoding service does not return geometries when the geocoded object is street-based, Google returns only a coordinate for geocoded object. This is a problem because it affects the quality of the results.

One common issue in this prototype application is the nature of Twitter text data, which includes abbreviated or hashtag locations (e.g. "Linha Vermelha" is also referred to as "#LinhaVermelha"). To address this issue, a synonym dictionary is used.

Another frequent issue involves classifying certain terms that define a region or a neighborhood. One example is *downtown* (in Portuguese, *centro* or *#centro*), which is often used as a direction but also as a reference. However, since routing services expect addresses or coordinates, this issue is handled by resorting to a particular database of general locations searched before any routing or geocoding operation is invoked.

Consider the following tweets as real examples:

(a) *"Acidente entre dois carros no Aterro do Flamengo".* (*"Accident between two cars at Aterro do Flamengo."*)
(b) *"Acidente envolvendo dois carros no Aterro do Flamengo, sentido #zonasul, na altura da Avenida Oswaldo Cruz."* (*"Accident involving two cars at Aterro do Flamengo, direction #zonasul, near Oswaldo Cruz Avenue."*)

The main location is always associated with a fact. To use this information, we referred to a specific dictionary to identify the type of fact and have a good visual representation of facts.

Figure 6 shows the results of the analysis of both tweets. Figure 11(A) illustrates the geocoding process without applying the techniques outlined in Section 5.1 (tweet (a)). Figure 11(B) shows the higher precision achieved by applying the techniques of Section 5.1 (tweet (b)), highlighting the correct side of street and the precise location of the accident.

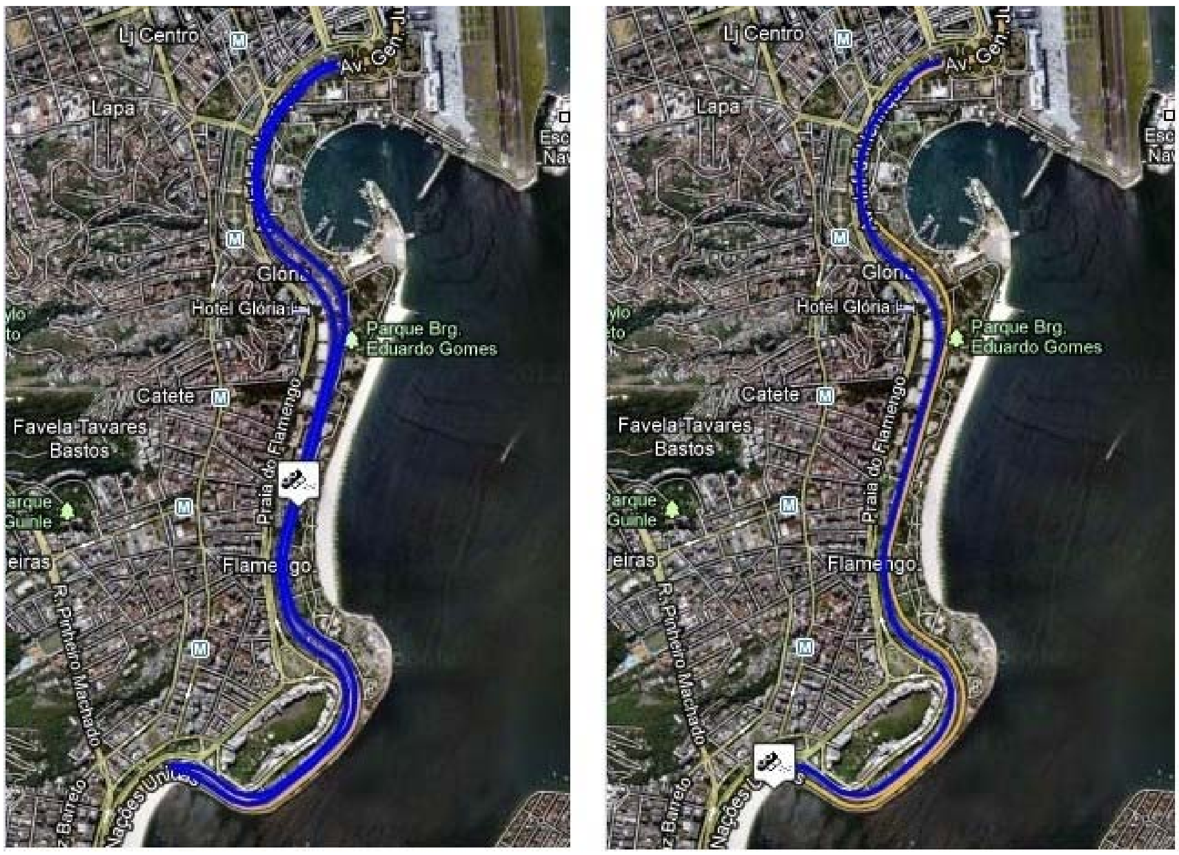


Figure 14 Locations extracted from the analysis of tweets

6.4. Prototype details

This section explains the components developed during the prototype implementation.

The component Environment Web Monitoring (EWM) was developed using the concepts and techniques explained in this dissertation.

- **Scrapper** – retrieves the traffic-related Twitter messages, using the API *Twitter4j*.
- **NLP (Natural Language Processor)** – analyses the news retrieved by the Scrapper and structures them. As mentioned in Section 5.4, it uses the F-EXT service to obtain the syntactic attributes of tokens.
- **Machine Learning** – executes all machine learning tasks, using some code interfaces and hiding from the final user all the implementation details of these algorithms. It is internally divided into two types of machine learning problems: Classification and Information Extraction.

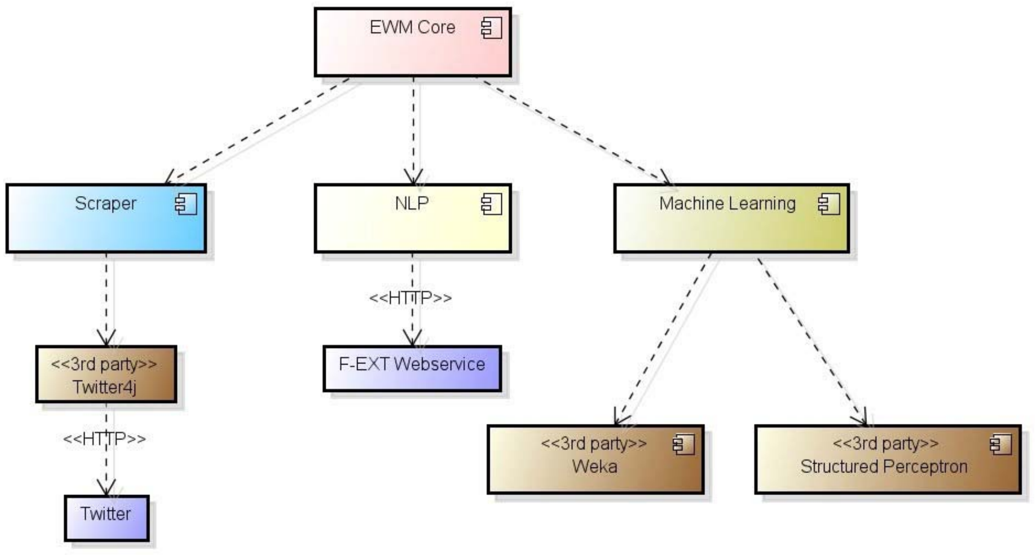


Figure 15 EWM overview

6.5. Chapter summary

This chapter described a solution that uses the structured traffic related message to automatically georeference the facts of interest with high precision and the EWM component. The location of fact has attributes like the side of road and reference points to improve the georeference results.

7
Conclusions and Future Work

In this dissertation, we first discussed basic requirements to achieve proactive monitoring of moving objects. Then, an architecture for proactive monitoring applications for moving objects is proposed. The first key point of the discussion was to model the process behind a moving object as a workflow to be able to infer future actions. The second key point was to monitor or even to predict changes in the environment by exploring dynamic data sources, such as Twitter.

This work proposed a structuring mechanism for traffic-related news without using predefined rules or templates. It is based only on machine learning and aims to identify and relate relevant information in the context of traffic, such as information on traffic flow status, accidents, road blocks, and their respective locations.

Using a corpus built for this work, the mechanism presented average f-measure for entity recognition of 90.41%, and accuracy of 73.37% with edge reduction, which helps reduce the number of possible connections among edges. Looking at Table 4, the FACT class recognition result of 60% was below the expected result of 70%, while all other results were satisfactory. This poor result is justified by the small corpus size, creating a precedent to increase the size of the corpus three times and to test the use of new features that can help identify possible events (e.g. verb and conjunction lists). This is left to future work.

The techniques presented are not restricted to Twitter and can be extended to data sources of different natures. One of the next steps is to test the proposed approach with tweets from ordinary users. The *@LeiSecaRJ* channel, for instance, publishes a series of tweets with idioms and excessive use of abbreviations.

As for future work we also plan to reduce the effort to build the corpus using active learning techniques, which offer various resources to optimize corpus construction. In our case, we plan to use the technique of extracting, from a much larger specialized news repository with around 35,000 pieces of news, the instances where the classifier indicates a low percentage difference between two entities. This way, the instances to be analyzed are those that create "doubts" to the classification algorithm, thus concentrating the manual efforts on examples with a high degree of uncertainty. On the other hand, unsupervised learning techniques can be used to reduce the cost of creating and maintaining a specific corpus for the problem.

Another future goal will be to use the Data entity, which was not used in this work because of the low number of occurrences in our corpus.

Using structured traffic news, a prototype application was built to georeference relevant facts over a road network. The prototype takes into account aspects of natural language regarding the description of the location of a fact. The initial results demonstrate that it is indeed possible to retrieve additional data from textual references and use them to improve the georeferencing task. The prototype can be used in applications that monitor moving vehicles in a road network in real time.

In the future, it is planned to use a cache strategy to reduce the network traffic overhead caused by the use of the Internet and to avoid exceeding the limits imposed by some Internet providers. Another goal is to automatically perform fact type inference, using thesaurus such as WordNet to parse facts from raw text.

A system that calculates alternative routes can be developed using the techniques developed in this dissertation as input to discover if some predefined route contains any traffic problem, such as blocked roads or traffic accidents..

The events generated by message structuration can be used as input for data mining applications to discover the most problematic routes.

To summarize, the contributions of this dissertation were: (a) an architecture for proactive monitoring applications; (b) a detailed description of how proactive monitoring applications work; (c) a mechanism to interpret traffic-related messages; (d) a geocoding model for traffic-related messages.

Four papers were already published as result of this dissertation: Chapter 3 was published as the paper “Proactive Monitoring of Moving Objects” [3], Chapter 5 as “Extrator de Fatos Relacionados ao Tráfego” [2], and Chapter 6 as “Georeferencing Facts in Road Networks” [4]. A summary of this dissertation was accepted for publication as the paper “A Proactive Application to Monitor Truck Fleets” [5].

8
References

[1] Aggrawal, C. C., 2007. Data Streams: Models and Algorithms. Ed. Springer. New York.

[2] Albuquerque, F. da C., Bacelar F. C., Tapia X. A. C., Carvalho M. T.: Extrator de Fatos Relacionados ao Tráfego. In Proceedings of the 27th Brazilian Symposium on Databases – SBBD 2012, São Paulo, Brazil, P. 169-176.

[3] Albuquerque F. da C., Barbosa I., Casanova M. A., Carvalho M. T., Macedo J. A.: Proactive Monitoring of Moving Objects, Proc. 14th International Conference on Enterprise Information Systems (ICEIS 2012, pages 191-194).

[4] Albuquerque F. da C., Barbosa I., Casanova M. A., Carvalho M. T., Georeferencing Facts in Road Networks. Proc. 13th Brazilian Symposium on GeoInformatics – GeoInfo 2012, Campos do Jordão, São Paulo, Brazil. pp. 120-127.

[5] Albuquerque F. da C., Barbosa I., Casanova M. A., Carvalho M. T., Macedo J. A. F., Renso C., A Proactive Application to Monitor Truck Fleets. Accepted at 14th International Conference on Mobile Data Management (MDM 2013) (http://mdm2013.dico.unimi.it/).

[6] Alvares, L. O., Loy, A. M., Renso, C., and Bogorny, V., 2011. An algorithm to identify avoidance behavior in moving object trajectories. In: Journal of the Brazilian Computer Society 11.

[7] Aquino J., Davis M., "JTS Topology Suite Technical Specifications, version 1.4", Vivid Solution, Inc., 2003

[8] Barbosa, I., Casanova, M. A., 2011. Trust Indicator for Decisions Based on Geospatial Data. In Geoinfo, XII International Symposium in GeoInformatics, pp. 49–60, Campos do Jordão.

[9] Bayegan, E., 1998. Mediators – (Intelligent) Middleware for Information Systems. Technical Report ISSN: 0802-6394 04/02, Norwegian University of Science and Technology, Department of Computer and Information Science, Trondheim.

[10] Borges K. A. V., Laender A. H. F., Medeiros C. B., Davis C. A.: Discovering geographic locations in web pages using urban addresses. GIR 2007: 31-36

[11] Bressan, S.; Cuiyu Zhang. "GéOO7: A Benchmark for XML Processing in GIS," Database and Expert Systems Applications, 2005. Proc. 16th International Workshop, pp.507-511, 26-26 Aug. 2005 doi: 10.1109/DEXA.2005.99

[12] Carvalho, S., Sarmento, L. and Rossetti, R. (2010). Real-Time Sensing of Traffic Information in Twitter Messages. In: ATSS @ IEEE ITSC 2010 Proceedings of 4th Workshop on Artificial Transportation Systems and Simulation, Madeira, Portugal

[13] Chen, Y. F., Di Fabbrizio, G., Gibbon, D., Jora, S., Renger, B., Wei, B., 2007. Geotracker: geospatial and temporal RSS navigation. In WWW'07, 16th International Conference on World Wide Web. pp. 41-50. Alberta.

[14] Chon, H. D., Agrawal, D. and Abbadi, A. E (2002). Query Processing for Moving Objects with Space-Time Grid Storage Model. In: MDM '02 Proceedings of the Third International Conference on Mobile Data Management.

[15] Chu, Y. J., Liu, T. H. (1965). On the shortest arbores-cence of a directed graph. Science Sinica(14), 1396--1400.

[16] Claro, D. B., Albers, P., Hao, J., 2006. Web Services Composition. in Processes and Applications, In: Cardoso, J., Sheth, A. (eds.): Semantic Web Services. pp. 205–234. Springer, New York.

[17] CloudMade, http://cloudmade.com

[18] CloudMade Java Library API, http://developers.cloudmade.com/projects/show/java-lib

[19] Earle, P., Bowden, D., Guy, M., 2011. Twitter earthquake detection: earthquake monitoring in a social world. In: Annals Of Geophysics, 54(6), [online] Available from: <http://www.annalsofgeophysics.eu/index.php/annals/article/viewFile/5364/5494> [Accessed January 19 2012].

[20] Edmonds, J. (1967). Optimum branchings. Journal of Research of the National Bureau of Standards(71B), 233-240.

[21] Endarnoto S. K., Pradipta S., Nugroho A. S., Purnama J.: Traffic Condition Information Extraction & Visualization from Social Media Twitter for Android Mobile Application, Proc. of 3rd International Conference on Electrical Engineering and Informatics (ICEEI 2011), CDROM H3-5, Institut Teknologi Bandung, Bandung, Indonesia, July 17-19, 2011

[22] Fritz, C., Kirschner, C., Reker, D., Wisplinghoff, A., Paulheim, H., Probst, F., 2010. Geospatial Web Mining for Emergency Management. In GIScience, Sixth international conference on Geographic Information Science. [online] Available from: <http://www.giscience2010.org/pdfs/paper_107.pdf> [Accessed January 10 2012].

[23] Geonames, [online] Available from: <www.geonames.org> [Accessed January 11 2012]

[24] GeoRSS, [online] Available from: <http://georss.org> [Accessed January 15 2012]

[25] Goldberg DW, Wilson JP, Knoblock CA: From Text to Geographic Coordinates: The Current State of Geocoding. URISA J 2007, 19(1):33-47. OpenURL

[26] Güting R. H. (2008). Algebras for Moving Objects and Their Implementation. geoinfoinfo, 26(1), p.1-7.

[27] Güting, R. H., de Almeida, V. T. and Ding, Z. (2006). Modeling and querying moving objects in networks. The VLDB Journal (2006) 15(2): 165–190.

[28] JTS Topology Suite, http://www.vividsolutions.com/jts

[29] Jung J. J., Online named entity recognition method for microtexts in social networking services: A case study of twitter, Expert Systems with Applications, Volume 39, Issue 9, July 2012, Pages 8066-8070, ISSN 0957-4174, 10.1016/j.eswa.2012.01.136. (http://www.sciencedirect.com/science/article/pii/S0957417412001546)

[30] Lewis, D. D.. Naive (bayes) at forty: The independence assumption in information retrieval. p. 4{15. Springer Verlag, 1998.

[31] MacEachren, A. M., Robinson, A. C., Jaiswal, A., Pezanowski, S., Savelyev, A., Blanford, J., Mitra, P., 2011. Geo-Twitter analytics: applications in crisis management, In 25th ICC, International Cartographic Conference, Paris. [online] Available from: <http://icaci.org/documents/ICC_proceedings/ICC2011/> [Accessed January 18 2012]

[32] Magoutas, B., Mentzas, G., Apostolou, D. (2011). Proactive Situation Management in the Future Internet: The Case of the Smart Power Grid. In: Proceedings of the 22nd International Workshop on Database and Expert Systems Applications, pp. 267-271.

[33] Mani, I. and Bloedorn E. (1998). Machine Learning of Generic and User-Focused Summarization. In Proceedings of the Fifteenth National Conference on Artificial Intelligence AAAI'98, pp. 821-826.

[34] Maron, M. E.. Automatic indexing: An experimental inquiry. J. ACM, 8:404{417, July 1961. 3.3

[35] Moreno, B., Times, V. C., Renso, C., and Bogorny, V., 2010. Looking inside the stops of trajectories of moving objects. In Geoinfo'10, XI Brazilian Symposium on Geoinformatics, pp. 9–20, Campos do Jordão.

[36] OGC, Open GeoSMS Standard - Core, 2012 [online] Available from: <http://www.opengeospatial.org/standards/opengeosms> [Accessed February 01 2012].

[37] Motta, E. N. , Fernandes, E. R., Milidiú, R. L.. F-EXT-2.0: A Web Service for Natural Language Processing. In: PROPOR 2010, software demonstration, Porto Alegre, RS, Brazil, April 27-30, 2010.

[38] Nadeau, David and Sekine, S. (2007) A Survey of Named Entity Recognition and Classification. In: Sekine, S. and Ranchhod, E.

Named Entities: Recognition, classification and use. Special issue of Lingvisticæ Investigationes. 30(1) pp. 3-26.

[39] Pfoser, D. and Jensen, C. S. (2001). Querying the Trajectories of On-Line Mobile Objects. In: MobiDE 2001 Proceedings of the 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access

[40] Platt, J. C. (1998). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, & A. Smola (Eds.). Advances in kernel methods: Support vector machines, Cambridge, MA: MIT Press.

[41] Prakash, S. S. S. and Kulkarni, M. N. (2003) Fleet Management: A GPS-GIS integrated approach In: Map India 2003 Proceedings of 6th Annual International Conference

[42] Ritter A., Clark S., Mausam, and Etzioni O. Named Entity Recognition in Tweets: An Experimental Study. Submitted, 2011.

[43] Sakaki, T., Okazaki, M., Matsuo, Y., 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In WWW'10, 19th International Conference on World Wide Web. ACM, New York.

[44] Santos, M. Y., and Moreira, A. (2010). GUESS: on the prediction of mobile users' movement in space, In: Wachowicz, M. (Ed.) Movement-Aware Applications for Sustainable Mobility: Technologies and Approaches, IGI Global Publishing.

[45] Siqueira, F. L. and Bogorny, V., 2011. Discovering chasing behavior in moving object trajectories. In Transactions in GIS, 15(5).

[46] Souza L. A., Delboni T M., Borges K. A. V., Davis C. A., Laender A. H. F.: Locus: Um Localizador Espacial Urbano. Proc. GeoInfo 2004: 467-478

[47] Tennenhouse, D., 2000. Proactive computing. In: Communications of the ACM, vol. 43, May. 2000, pp. 43–50.

[48] Tsakiri, M., Stewart, M., Forward, T. Sandison, D. and Walker, J., 1998. Urban fleet monitoring with GPS and GLONASS. In: System, 51(3), p.382-393.

[49] Vieira, T.A.S.C., Casanova, M.A. and Ferrão, L.G., 2004. An Ontology-driven Architecture for Flexible Work?ow Execution. In LA-Web 2004 Joint Conference 10th Brazilian Symposium on Multimedia and the Web 2nd Latin American Web Congress, pp.70-77. Ribeirão Preto.

[50] Weka, http://www.cs.waikato.ac.nz/ml/weka

9 APPENDIX A – ADDITIONAL RESULTS

9.1. Named Entity Recognition

These are other results obtained during the development of this work:

Features: CurrT(W)

SMO - 1175.134 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.0503	0	0	0	0	0	0	0	0	0
COREF_LOC	99.9275	97.9615	93.3974	95.3276	4.825	93.3974	100	93.3974	96.3532	5.0238
DIR_N	99.5406	96.6477	88.4141	92.1621	7.7722	87.3191	99.7297	87.4403	92.8173	6.7987
DIR_NS	99.7187	93.9394	76.4394	82.8868	7.981	20.6362	24.7856	31.5183	34.1964	28.7837
FACT	98.1423	85.5475	66.2292	73.629	5.3353	31.3531	57.1097	38.5135	44.5479	21.8074
LOC	91.507	86.6678	81.8503	83.2813	3.3491	31.4376	47.5126	47.9457	47.3738	8.4901
O	87.2882	85.6552	92.5318	88.7586	1.8723	92.5318	100	92.5318	96.0276	3.1381
OPEN	99.8052	95.8333	81.0444	86.3089	14.7514	72.6411	91.1818	77.8998	81.1818	19.8806
REF_N	99.8383	98.9118	93.1938	95.8182	3.8512	92.2472	100	92.2472	95.7983	4.2567
TI_1	98.9304	67.9119	51.5681	56.8851	10.5759	5.2133	10.6389	9.8254	15.7511	6.6374
TI_2	98.509	73.5607	86.7414	79.33	5.2407	43.4933	54.7685	67.2217	60.298	6.637
TI_3	98.984	76.8021	55.3094	61.2824	10.5654	15.0623	20.9178	25.6943	25.4222	21.4504
Média	97.7803	78.4184	70.5169	72.3079	7.0627	47.4871	57.8829	54.1720	57.9431	12.3453

Naïve Bayes - 2.337 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.0503	0	0	0	0	0	0	0	0	0
COREF_LOC	99.9358	98.5962	93.3974	95.6558	5.0111	93.3974	100	93.3974	96.3532	5.0238
DIR_N	99.5315	98.0966	86.6159	91.4664	8.9322	85.6733	99.8649	85.7339	91.6081	9.0152
DIR_NS	99.6319	96.9697	66.4015	77.2805	9.5771	10.3181	12.3928	15.7592	34.1964	28.7837
FACT	98.0144	91.1152	59.0831	69.9456	10.9557	28.8015	56.6027	34.6476	41.4713	22.3051
LOC	92.3028	91.192	79.4506	84.2784	2.876	29.1378	46.1372	43.8897	44.7023	8.1497
O	87.0553	83.4849	95.4377	88.8822	1.5548	95.4377	100	95.4377	97.5956	2.7305
OPEN	99.6513	82.9167	53.7614	60.9481	24.9643	48.7618	94.8128	51.3912	67.4374	26.9189
REF_N	99.8346	99.4559	92.5275	95.69	4.2725	91.5955	100	91.5955	95.4082	4.7242
TI_1	98.9304	78.956	41.5164	50.9001	14.5382	4.5892	14.384	8.2095	15.1377	7.0122
TI_2	98.5616	74.8025	86.2505	79.8532	5.1455	42.9818	54.4247	66.6524	59.8586	6.0241
TI_3	99.0643	88.4011	52.5376	63.5077	10.3609	12.9871	18.7245	22.5091	23.8532	19.8025
Média	97.8040	77.9990	63.9984	67.7365	8.7603	43.0524	57.1675	48.4018	56.2395	12.9288

Features: CurrT(W), CurrT(SW), PrevT(W, 2), NextT(W, 2), PrevT(POS, 2), CurrT(POS), NextT(POS, 2)

SMO - 1248.508 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.8389	81.5913	86.6569	83.7308	4.1259	86.6569	100	86.6569	98.0562	2.3032
COREF_LOC	99.9387	97.6197	96.7308	97.0423	3.3825	96.7308	100	96.7308	98.2561	2.9519
DIR_N	99.5506	94.236	90.6431	92.2246	8.4277	89.7509	100	89.7509	94.1556	7.1296
DIR_NS	99.991	100	98.75	99.3333	2	92.5	93.3333	95	94	18
FACT	98.6746	87.1675	78.5947	82.2047	5.6518	41.1494	63.598	51.3378	55.5257	18.9894
LOC	98.0852	96.0293	96.5068	96.249	1.3045	81.956	90.1988	89.6983	89.919	4.2415
O	95.5596	95.1923	96.8035	95.9847	0.7214	96.8035	100	96.8035	98.3725	0.5781
OPEN	99.8244	95.8333	81.6959	87.3975	10.9967	73.3972	91.3333	77.581	82.1486	18.0706
REF_N	99.8115	97.4306	92.924	94.9154	4.3037	92.0066	100	92.0066	95.6239	4.7862
TI_1	99.807	96.0833	90.6964	92.7983	4.3274	65.2113	84.9206	70.3363	75.1927	22.9188
TI_2	99.7656	96.8455	96.4768	96.5867	2.1926	83.507	92.6066	89.4833	90.7115	5.6546
TI_3	99.8009	97.0492	94.1375	95.318	5.4792	85.484	96.0027	87.875	90.7107	13.5972
Média	99.2751	92.8522	88.8936	90.2912	5.1806	78.5246	90.9867	81.9175	86.0389	11.1847

Naïve Bayes - 10.025 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.4636	64.9559	86.5343	71.7682	18.3238	86.5343	100	86.5343	97.9935	2.1088
COREF_LOC	99.7916	85.9071	94.4071	89.3108	12.1819	94.4071	100	94.4071	96.954	4.2702
DIR_N	99.3131	88.554	87.2439	87.3168	10.2179	86.3727	100	86.3727	92.1252	8.112
DIR_NS	99.8841	92.6624	97.4583	94.6185	9.3446	89.7727	92.2917	92.2917	92.125	19.1911
FACT	97.3674	69.1008	79.6229	71.7656	13.2846	36.3517	52.6677	50.1803	49.9356	21.79
LOC	96.7043	93.8938	93.2336	93.517	3.4075	66.7925	78.8877	78.9368	78.878	12.1577
O	92.0579	94.3053	91.0719	92.5472	3.999	91.0719	100	91.0719	95.1904	3.8366
OPEN	99.6823	83	69.7505	73.502	19.1671	62.8377	92.3333	65.7093	73.9542	20.5076
REF_N	99.5297	86.7728	93.6563	89.1464	10.9587	92.4612	99.6875	92.6805	95.862	4.8786
TI_1	99.3875	79.5389	76.3751	76.1357	22.4764	43.4057	61.7962	50.1442	59.7674	26.8
TI_2	98.8706	80.672	95.811	86.4154	11.999	77.9216	85.9235	88.1669	86.7365	10.1319
TI_3	99.471	82.2677	89.9345	83.9992	16.7548	73.7234	84.1799	79.0976	85.1603	16.6266
Média	98.5698	81.1639	84.6230	81.0033	13.1891	71.7553	85.9089	76.1354	82.0696	13.6724

Features: CurrT(W), CurrT(SW), PrevT(W, 2), NextT(W, 2), PrevT(POS, 2), CurrT(POS), NextT(POS, 2), LocType, CurrWSC, User

SMO - 1190.052 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.8389	81.5913	86.6569	83.7308	4.1259	86.6569	100	86.6569	98.0562	2.3032
COREF_LOC	99.9387	97.6197	96.7308	97.0423	3.3825	96.7308	100	96.7308	98.2561	2.9519
DIR_N	99.5506	94.236	90.6431	92.2246	8.4277	89.7509	100	89.7509	94.1556	7.1296
DIR_NS	99.991	100	98.75	99.3333	2	92.5	93.3333	95	94	18
FACT	98.7012	87.5066	78.7355	82.4627	5.5461	41.4152	64.0338	51.5378	55.8494	18.7878
LOC	98.1967	96.3009	96.6369	96.4513	1.3462	83.0166	91.2361	90.0075	90.5755	3.9688
O	95.6803	95.2778	96.9464	96.0995	0.729	96.9464	100	96.9464	98.4464	0.563
OPEN	99.8334	92.5	83.855	86.9956	8.9815	75.0965	91.5714	79.3171	83.6777	16.3823
REF_N	99.8115	97.4306	92.924	94.9154	4.3037	92.0066	100	92.0066	95.6239	4.7862
TI_1	99.807	96.0833	90.6964	92.7983	4.3274	65.2113	84.9206	70.3363	75.1927	22.9188
TI_2	99.7656	96.8455	96.4768	96.5867	2.1926	83.507	92.6066	89.4833	90.7115	5.6546
TI_3	99.8091	97.5037	94.1375	95.5042	5.2367	85.484	96.0027	87.875	90.7107	13.5972
Média	99.2963	92.6843	89.0915	90.3188	5.0025	78.7684	91.1183	82.1012	86.2376	11.0172

Naïve Bayes - 7.84 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.4636	64.9559	86.5343	71.7682	18.3238	86.5343	100	86.5343	97.9935	2.1088
COREF_LOC	99.7576	83.3992	94.4071	87.7758	12.4404	94.4071	100	94.4071	96.954	4.2702
DIR_N	99.3404	89.2002	87.2439	87.6747	10.1499	86.3727	100	86.3727	92.1252	8.112
DIR_NS	99.8845	92.1299	98.0833	94.5848	8.3058	90.1894	91.7917	93.9583	92.625	17.9143
FACT	97.3518	68.9352	79.3253	71.6087	13.4403	36.0707	52.481	49.7374	49.58	22.0174
LOC	96.846	94.7566	92.823	93.731	3.3042	65.8266	78.4611	77.4256	77.8993	13.5267
O	92.089	94.0768	91.3701	92.6062	4.0091	91.3701	100	91.3701	95.362	3.7122
OPEN	99.6827	81.0833	69.8301	72.6343	18.472	62.6873	92.4524	65.5773	74.0243	19.8513
REF_N	99.5343	86.9611	93.6563	89.2222	11.2223	92.4612	99.6875	92.6805	95.862	4.8786
TI_1	99.3874	80.0212	75.637	75.922	22.1009	42.3197	59.7327	48.9354	58.0358	28.6143
TI_2	98.8347	80.136	95.811	86.0246	12.3213	78.1915	86.2253	88.2156	86.9224	10.0149
TI_3	99.4679	82.1334	90.3618	84.0897	16.6614	73.9615	84.253	79.4101	85.3461	16.548
Média	98.5787	80.8684	84.6218	80.8186	13.0842	71.6584	85.7025	76.0609	81.9194	13.7614

Features: CurrT(W), CurrT(SW), CurrT(STW) PrevT(W, 2), NextT(W, 2), PrevT(STW, 1), NextT(STW, 1), PrevT(POS, 2), CurrT(POS), NextT(POS, 2), LocType, CurrWSC

SMO - 878.467 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.821	80.703	82.5543	81.4272	8.9802	91.727	100	91.727	95.3895	5.8351
COREF_LOC	99.9477	97.6197	97.5641	97.5166	2.7655	97.5641	100	97.5641	98.7304	1.9414
DIR_N	99.5756	95.0212	90.5576	92.5441	7.7553	89.6625	100	89.6625	94.1274	6.9168
DIR_NS	99.991	100	98.75	99.3333	2	92.5	93.3333	95	94	18
FACT	98.657	86.3581	79.4841	82.2541	6.8042	42.1908	63.7343	52.4614	56.112	20.6829
LOC	98.2163	96.3644	96.6628	96.4993	1.2128	82.6255	90.6882	90.0362	90.3315	4.1279
O	95.6968	95.3772	96.8573	96.1057	0.7531	96.8573	100	96.8573	98.4003	0.5766
OPEN	99.845	92.6282	87.6732	89.5995	8.8236	76.6999	89.9318	82.7401	84.8734	15.5431
REF_N	99.8115	97.4306	92.924	94.9154	4.3037	92.0066	100	92.0066	95.6239	4.7862
TI_1	99.7915	92.5195	91.0096	91.5237	8.1711	69.7823	88.9615	72.2288	78.1724	23.927
TI_2	99.7822	96.8676	96.9707	96.8395	2.1428	85.6517	93.9109	90.6236	91.9195	6.1331
TI_3	99.8452	98.4117	94.8555	96.3766	4.4968	87.7968	98.3766	89.048	92.374	11.7947
Média	99.3007	89.6386	87.1818	88.0719	5.5879	80.7421	91.5208	84.0030	86.6067	11.2650

Naïve Bayes - 8.704 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.6803	74.7394	74.6002	73.9389	15.0436	82.8891	100	82.8891	90.0299	8.3636
COREF_LOC	99.8101	87.2138	94.4071	90.1208	11.9316	94.4071	100	94.4071	96.954	4.2702
DIR_N	99.3816	90.2459	87.5154	88.5784	9.067	86.6375	100	86.6375	92.3472	7.5446
DIR_NS	99.8105	93.6965	87.2614	89.8435	12.1453	58.7297	62.0119	68.1629	NaN	36.2661
FACT	97.4917	69.4466	80.9333	72.6852	12.8982	38.3928	55.1253	52.1603	52.1868	21.3072
LOC	97.0637	94.741	93.777	94.2113	2.9923	68.2459	79.9786	80.0734	79.9874	11.7677
O	92.6988	94.0971	92.5146	93.2399	3.4348	92.5146	100	92.5146	96.0253	3.0418
OPEN	99.7099	87.9312	71.0628	76.8511	20.3891	62.6201	91.8707	66.4899	73.7439	20.6248
REF_N	99.5683	87.8302	93.6563	89.7542	10.7438	92.4612	99.6875	92.6805	95.862	4.8786
TI_1	99.446	80.0422	74.1002	75.3825	25.1476	48.4589	78.478	50.2832	NaN	27.7916
TI_2	98.9391	81.2576	96.3079	87.0708	11.5541	79.578	87.0406	89.325	87.8615	9.5577
TI_3	99.6363	88.0205	90.6493	87.4096	15.8883	76.9068	88.317	82.299	84.2762	19.5314
Média	98.7009	80.6868	80.9066	79.6550	13.3733	70.5776	84.5520	75.3530	84.9274	15.4712

Features: CurrT(W), CurrT(SW), CurrT(STW) PrevT(W, 2), NextT(W, 2), PrevT(STW, 1), NextT(STW, 1), PrevT(POS, 2), CurrT(POS), NextT(POS, 2), LocType, CurrWSC, CurrT(FactLike), PrevT(FactLike, 1), NextT(FactLike, 1), PrevT(SMW, 1), NextT(SMW, 1)

SMO - 878.467 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.8469	81.7387	87.4902	84.1478	4.5624	97.21	100.00	97.21	98.54	2.20
COREF_LOC	99.9477	97.6197	97.5641	97.5166	2.7655	97.56	100.00	97.56	98.73	1.94
DIR_N	99.5662	94.3595	90.8701	92.3602	8.1589	89.91	99.72	90.03	94.24	7.13
DIR_NS	99.9910	100	98.7500	99.3333	2	92.50	93.33	95.00	94.00	18.00
FACT	98.7401	88.8560	82.2401	85.0976	5.8015	45.51	70.74	55.01	60.62	16.84
LOC	98.2575	96.4089	96.8178	96.5933	1.1355	86.69	94.43	91.28	92.79	2.90
O	95.7569	95.4931	96.7931	96.1321	0.9151	78.29	91.75	82.67	85.60	16.97
OPEN	99.8789	98.5714	87.1212	92.0097	8.3971	92.01	100.00	92.01	95.62	4.79
REF_N	99.8024	96.9544	92.9240	94.6939	4.3382	70.11	87.78	74.90	79.76	18.70
TI_1	99.7981	96.3839	90.3846	92.7799	4.2982	90.11	96.87	92.79	94.66	3.79
TI_2	99.7725	96.6315	96.9707	96.7099	2.2335	87.57	99.33	87.87	92.06	12.68
TI_3	99.8181	98.0593	94.1375	95.7221	4.8241	85.41	94.50	87.82	90.45	8.88
Média	99.3151	92.5443	89.7741	90.4946	3.8023	97.21	100.00	97.21	98.54	2.20

Naïve Bayes - 8.704 segundos										
	By Token					Set of Tokens				
Classes	Acc	Prec	Recall	F1	St. Dev. F1	Acc	Prec	Recall	F1	St. Dev. F1
ABSL_N	99.5625	68.4262	85.9902	74.1746	17.2088	95.5447	100	95.5447	97.6305	3.1091
COREF_LOC	99.7512	83.049	94.4071	87.4614	14.4025	94.4071	100	94.4071	96.954	4.2702
DIR_N	99.3237	88.0481	88.0728	87.7033	10.1854	87.3158	99.8611	87.3769	92.7341	7.6235
DIR_NS	99.7886	86.4615	97.4583	90.5703	13.023	89.7727	92.7381	91.9444	92.125	19.1911
FACT	97.7727	75.481	83.4098	77.7066	11.0264	44.9297	67.6428	55.4305	59.7448	18.4593
LOC	96.9288	94.3849	93.6339	93.9593	3.2862	71.9983	83.8053	81.6644	82.6812	11.1347
O	92.3052	94.8359	90.8887	92.6793	4.2141	92.5368	100	92.5368	96.0212	3.3025
OPEN	99.6933	83.4048	74.3977	77.0707	21.8073	69.6024	94.21	72.431	78.8427	20.3421
REF_N	99.471	85.3146	93.6563	88.157	11.9955	92.4612	99.6875	92.6805	95.862	4.8786
TI_1	99.3126	75.9094	78.8168	75.3735	22.9765	52.9482	74.4258	57.8058	63.6544	21.2081
TI_2	98.8649	80.3076	97.1355	86.6582	11.9484	88.917	94.61	93.4799	93.9395	4.5161
TI_3	99.3197	78.4901	91.0743	81.4621	20.6233	80.3718	91.9011	83.794	86.8402	17.957
Média	98.5079	82.8428	89.0785	84.4147	13.5581	80.0671	91.5735	83.2580	86.4191	11.3327

9.2. Information Extraction

Results without edge reduction:

Set of features in each execution	Accuracy by edge	Accuracy by example	Standard deviation of accuracy by example	Time spent in training and testing (seconds)
PossRel. ConcT(NE). ConcT(W)	77.392	37.5655	6.5768	3.077
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE)	87.0167	53.5106	7.7601	8.57
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.1). NearT(W.1). NearT(NE.1)	89.3037	64.6645	5.5735	11.98
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.2). NearT(W.2). NearT(NE.2)	90.4879	64.8773	8.0945	13.711
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.3). NearT(NE.3)	89.9049	65.3028	8.8845	14.817
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.4). NearT(W.4). NearT(NE.4)	90.1930	65.6383	8.4579	15.782
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2)	90.8314	67.2586	10.8354	13.571
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2). EntDist(RE)	90.7264	66.4075	8.203	13.479
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2). AbsLocPair. MetaWithLoc. BetT(PUNCT)	92.0968	68.5147	8.9624	14.888

Results with edge reduction:

Set of features in each execution	Accuracy by edge	Accuracy by example	Standard deviation of accuracy by example	Time spent in training and testing (seconds)
PossRel. ConcT(NE). ConcT(W)	77.448	37.9705	7.3175	2.466
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE)	87.6586	56.9763	7.4951	4.056
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.1). NearT(W.1). NearT(NE.1)	89.8410	64.3412	9.3892	5.38
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.2). NearT(W.2). NearT(NE.2)	90.5869	66.7021	11.1377	6.0150
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.3). NearT(NE.3)	90.4021	66.7226	10.8767	6.395
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.4). NearT(W.4). NearT(NE.4)	90.2813	67.8764	8.8432	6.96
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2)	91.0562	67.32	10.3451	6.152
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2). EntDist(RE)	90.9551	67.1686	11.6224	6.447
PossRel. ConcT(NE). ConcT(W). BetT(RE). BetT(W). BetT(NE). NearT(RE.3). NearT(W.2). NearT(NE.2). AbsLocPair. MetaWithLoc. BetT(PUNCT)	92.6961	73.2651	10.6006	6.124