



Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind



A methodology for traffic-related Twitter messages interpretation

Fábio C. Albuquerque^a, Marco A. Casanova^a, Hélio Lopes^{a,*}, Luciana R. Redlich^a,
José Antonio F. de Macedo^b, Melissa Lemos^c, Marcelo Tilio M. de Carvalho^c, Chiara Renso^d

^a Dep. de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Brazil

^b Dep. de Computação, Universidade Federal do Ceará, Brazil

^c Instituto TecGraf, Pontifícia Universidade Católica do Rio de Janeiro, Brazil

^d Inst. di Scienza e Tecn. dell'Info., CNR at Pisa, Italy

ARTICLE INFO

Article history:

Received 1 December 2014

Received in revised form 8 October 2015

Accepted 13 October 2015

Available online xxx

Keywords:

Twitter analysis

Natural language processing

Traffic monitoring

ABSTRACT

This paper addresses the problem of interpreting tweets that describe traffic-related events and that are distributed by government agencies in charge of road networks or by news agencies. Processing such tweets is of interest for two reasons. First, albeit phrased in natural language, such tweets use a much more regular and well-behaved prose than generic user-generated tweets. This characteristic facilitates automating their interpretation and achieving high precision and recall. Second, government agencies and news agencies use Twitter channels to distribute real-time traffic conditions and to alert drivers about planned changes on the road network and about future events that may affect traffic conditions. Hence, such tweets provide exactly the kind of information that proactive truck fleet monitoring and similar applications require. The main contribution of the paper is an automatic tweet interpretation tool, based on Machine Learning techniques, that achieves good performance for traffic-related tweets distributed by traffic authorities and news agencies. The paper also covers in detail experiments with real traffic-related tweets to access the precision and recall of the tool.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

According to Tennenhouse [1], to be proactive, an application must detect interesting situations before they happen and must be able to handle such situations without human supervision. In particular, to achieve proactive behavior, an application that monitors moving objects must: (1a) model the behavior of the moving objects; (1b) monitor the current state of the objects; (2a) model the environment where the objects move; (2b) monitor the current state of the environment; (3a) detect environment changes that may affect the future behavior of the moving objects; and (3b) adjust the planned behavior of the moving objects to the changes. Inexpensive positioning devices and communication technologies cater for (1b), whereas Web resources, notably Twitter channels, RSS and geoRSS [2] feeds and Open GeoSMS [3], provide valuable

data from which to extract event descriptions that affect the environment where the objects move.

In this paper, we focus on the problem of interpreting tweets that describe traffic-related events and that are tweeted by government agencies in charge of road networks and by news agencies. Processing such tweets is of interest for essentially two reasons. First, albeit phrased in natural language, such tweets use a much more regular and well-behaved prose than generic user-generated tweets. This characteristic facilitates automating their interpretation and achieving high precision and recall. Second, government agencies and news agencies use Twitter channels to distribute real-time traffic conditions and to alert drivers about planned changes on the road network and about future events that may affect traffic conditions. Hence, such tweets provide real-time or future information about the road network, which is exactly the kind of information that proactive truck fleet monitoring and similar applications require.

Contributions. The first contribution of the paper is a domain ontology, called TEDO, that models traffic-related situations as *events*, composed of actors, locations and timestamps. The main contribution of the paper is an automatic tweet interpretation tool, based on Machine Learning techniques, that achieves good

* Corresponding author.

E-mail addresses: fabuquerque@inf.puc-rio.br (F.C. Albuquerque), casanova@inf.puc-rio.br (M.A. Casanova), lopes@inf.puc-rio.br (H. Lopes), lredlich@gmail.com (L.R. Redlich), jose.macedo@lia.ufc.br (J.A.F. de Macedo), melissa@tecggraf.puc-rio.br (M. Lemos), tilio@tecggraf.puc-rio.br (M.T.M. de Carvalho), chiara.renso@isti.cnr.it (C. Renso).

<http://dx.doi.org/10.1016/j.compind.2015.10.005>

0166-3615/© 2015 Elsevier B.V. All rights reserved.

performance for traffic-related tweets distributed by traffic authorities and news agencies. In particular, given a traffic-related tweet, the tool uses named-entity recognition techniques to identify the location of the event the tweet describes and relation extraction methods to capture relations between the components of the event. The tool transforms each such tweet into a set of RDF triples [4], constructed according to the TEDO ontology. Finally, the paper covers in detail experiments with real traffic-related tweets, which indicate that the tool achieves high precision and recall.

Paper outline. Section 2 introduces the TEDO ontology. Section 3 defines the notions of tagged tweets and dependency trees and illustrates how to represent of a tweet in TEDO. Section 4 describes some implementation details and the experiments with the tool. Section 5 discusses related work. Finally, Section 6 presents the conclusions.

2. TEDO – a traffic event domain ontology

This section introduces the *Traffic Event Domain Ontology*, TEDO, that models traffic-related situations as *events*, composed of actors, locations and timestamps. Section 2.1 briefly reviews some ontology concepts, while Section 2.2 covers the details of the classes and properties of TEDO. Section 3.5 contains an example of a tweet represented in TEDO.

TEDO is generically based on the notion of event [5] and relations between events [6,7]. This design decision reflects the principle that traffic is a process modeled by inter-related discrete events. TEDO also borrows some concepts from two traffic accident ontologies [8,9].

The development of TEDO used two major sources to induce some of the property values. The first source was a gazetteer, which provided the names and coordinates of the locations that populate the class Location (see Section 2.2). The second source was a tweet corpus (see Section 4.2), which induced the values of some of the datatype properties (see Tables 1 and 4).

2.1. A brief review of some ontology concepts

A class *C* is a set of instances or individuals. Rather than referring to “an instance of class *C*” we will simply say “a *c*”. For example, instead of “an instance of class Accident”, we say “an accident”. We may also declare that a class *D* is a subclass of *C* to indicate that all instances of *D* are also instances of *C*.

We will use XML Schema *simple types*, such as *string*, *float*, *boolean* and *dateTime*, *enumeration types* that define a list of values and *complex types* created by combining the simple types [10]. We will refer to an XML Schema type simply as a *type*.

A *datatype property* *P* is a binary relation between the set of instances of a class *D* and the set of values of a type *T*; we say that *D* is the *domain* of *P* and *T* is the *range* of *P*. Datatype properties capture in OWL the equivalent of attributes in UML or in the entity-relationship model, familiar to database designers.

An *object property* *O* is a binary relation between the set of instances of a class *D* and the set of instances of a class *R*; we say that *D* is the *domain* of *O* and *R* is the *range* of *O*. Object properties capture in OWL the equivalent of relationships in UML or binary relationships in the entity-relationship model. However, OWL does not offer constructs for relations with attributes or *n*-ary relations, with *n* > 2. In such situations, the designer has to resort to the *reification* of the relation [11].

A *cardinality restriction* for a class *C* imposes limitations on the number of occurrences of a datatype or object property *Q* each instance of *C* must have. A *maximum* (or *minimum*) cardinality restriction specifies the maximum (or minimum) number of occurrences of property *Q* each instance of *C* must have. Departing from OWL and adopting UML notation for cardinalities, we use “*m*..*n*” to indicate that the number of occurrences of property *Q* for an instance of *C* must be at least *m* and at most *n*. Furthermore, when *m* is “0”, *Q* may not be defined for some instances of *C* and, when *n* is “*”, *Q* may associate an instance of *C* with an unbounded number of instances of the range of *Q*.

Finally, an *international resource identifier* (IRI) [4] identifies a resource. The notion of IRI is a generalization of URI (Uniform Resource Identifier), allowing non-ASCII characters to be used in the IRI character string. We will store the result of analyzing a tweet as a set of *RDF triples* [4] of the form (*s*, *p*, *o*), where *s* is the *subject*, *p* is the *predicate* and *o* is the *object* of the triple. The subject and the predicate are IRIs and *o* is either an IRI or an XML literal.

2.2. Classes and properties of TEDO

This section describes the classes and properties of TEDO, summarized in Fig. 1 and in Tables 1–3 (the last two also include cardinality restrictions).

Traffic events: Traffic events in TEDO are instances of the class TrafficEvent, which is specialized into the following subclasses (see Fig. 1), with the intended interpretation of their instances:

- Interdiction: an interdiction that affects the traffic;
- Accident: an accident involving one or more vehicles, such as a collision, that affects the traffic;
- Breakdown: a vehicle breakdown that affects the traffic;

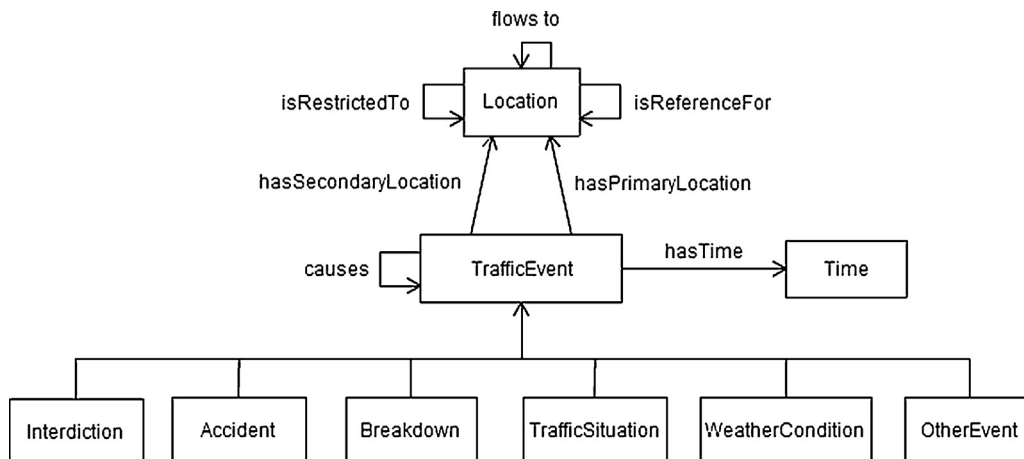


Fig. 1. TEDO classes and object properties (datatype properties omitted for legibility).

Table 1
Summary of TEDO classes.

Class name	Description	Super class
TrafficEvent	A traffic event	
Interdiction	An interdiction	TrafficEvent
Accident	An accident	TrafficEvent
Breakdown	A vehicle breakdown	TrafficEvent
TrafficSituation	A traffic situation	TrafficEvent
WeatherCondition	A traffic event caused by weather conditions	TrafficEvent
OtherEvent	An unclassified traffic event	TrafficEvent
Location	A place related to a traffic event	
Time	A placeholder for traffic event timestamps	

- **TrafficSituation**: an observed traffic situation, characterized by an *intensity* (see the definition of the datatype property `hasTrafficIntensity` below), expressed in a tweet such as “Traffic at Praça Sibelius has *nearly stopped*”;
- **WeatherCondition**: an event, caused by weather conditions, such as rain or flooding, that affects the traffic;
- **OtherEvent**: an event, other than those listed above, such as a defective spotlight, that affects the traffic.

Actor, location and time of a traffic event: Traffic-related tweets typically indicate one or more locations for the events they describe, the time of the events and the actors (or agents) involved in the events. For example, the tweet “Accident between 2 cars at Av das Américas in lane direction Grota Funda near number 19880”, posted on March 5th, 2012 at 07:07:01, indicates: a traffic event of the class `Accident`; the actors (“between 2 cars”) of the accident; the *primary location* (“Av das Américas”) of the accident; a *secondary location* (“Grota Funda”) defining the flow of direction of the traffic lanes affected (“in lane direction”) by the accident; a secondary location giving a reference point (“near Number 19880”) for the accident.

Class `TrafficEvent` is the domain of a datatype property, `hasActor`, and three object properties, `hasPrimaryLocation`, `hasSecondaryLocation` and `hasTime`, discussed in what follows.

Table 2
Summary of TEDO object properties.

Object property	Description	Card.	Domain	Range
<code>hasPrimaryLocation</code>	Associates a traffic event with a primary location	0..1	<code>TrafficEvent</code>	<code>Location</code>
<code>hasSecondaryLocation</code>	Associates a traffic event with a secondary location	0..n	<code>TrafficEvent</code>	<code>Location</code>
<code>hasTime</code>	Associates a traffic event with an instance of <code>Time</code>	0..1	<code>TrafficEvent</code>	<code>Time</code>
<code>flowsTo</code>	Relates two locations, L and M, such that L is a two-way road and M indicates the flow of direction of the traffic lanes of L affected by a traffic event	0..1	<code>Location</code>	<code>Location</code>
<code>isReferenceFor</code>	Relates two locations, M and L, such that M is a reference point for a traffic event that occurred in L	0..*	<code>Location</code>	<code>Location</code>
<code>isRestrictedTo</code>	Relates two locations, L and M, such that L is restricted to the area defined by M	0..*	<code>Location</code>	<code>Location</code>
<code>causes</code>	Indicates that a source event is a cause for a target event	0..*	<code>TrafficEvent</code>	<code>TrafficEvent</code>

Table 3
Summary of TEDO datatype properties.

Datatype property	Description	Card.	Domain	Range
<code>hasActor</code>	An actor of a traffic event	0..*	<code>TrafficEvent</code>	string
<code>hasLocationName</code>	A name for a location	0..*	<code>Location</code>	string
<code>hasCoordinates</code>	The coordinates of a location	0..1	<code>Location</code>	(complex)
<code>affectsBothDirections</code>	True, if the event affects the traffic flow in both directions of a two-way road; and False, otherwise	0..1	<code>Location</code>	boolean
<code>hasPostingTime</code>	The unique timestamp when the tweet was posted	0..1	<code>Time</code>	dateTime
<code>hasPublicationTime</code>	The unique timestamp when the news was published	0..1	<code>Time</code>	dateTime
<code>hasEventTime</code>	A unique noun phrase indicating when the traffic event occurred or will occur	0..1	<code>Time</code>	string
<code>hasTrafficIntensity</code>	Traffic intensity at some point in time	0..1	<code>TrafficSituation</code>	G, H, S

Actor of a traffic event: Traffic-related tweets typically indicate the actors of the events using a multitude of terms, such as “car”, “motorcycle”, “truck” and “pedestrian”. Because of this richness, in TEDO, actors are described merely as strings using the following property:

- `hasActor` (with `TrafficEvent` as domain and the type string as range): describes an actor of an event.

In future developments, we plan to create an `Actor` class, along with its subclasses, as pointed out in the conclusions.

Location of a traffic event: Traffic-related tweets express the place where an event occurred in a variety of ways, often with a richness of details that helps users locate the event. A tweet typically refers to: a street name or a neighborhood; a building or a facility, such as an airport or a shopping mall; or a specific kilometer on a highway, a road exit, or any reference location, such as “km 135” or “exit 5”. To describe traffic-related places, TEDO has a class:

- `Location`: a place related to a traffic event; and two object properties:
- `hasPrimaryLocation` (with `TrafficEvent` as domain and `Location` as range): associates a traffic event with a single location, designated as the primary location of the event;
- `hasSecondaryLocation` (with `TrafficEvent` as domain and `Location` as range): associates a traffic event with one or more secondary locations that help describe where the event occurred.

To qualify a location, TEDO has the following datatype properties:

- `hasLocationName` (with `Location` as domain and the type string as range): one or more names for a location, either primary or secondary;
- `hasCoordinates` (with `Location` as domain and a complex type `coordinatePair` as range): the unique coordinates of a location,

expressed as a pair of strings describing the latitude and longitude of the location, either primary or secondary.

Furthermore, to model complex descriptions of the location of a traffic event (see Section 4.1.2 for examples), TEDO has the following datatype property:

- **affectsBothDirections** (with Location as domain and the type boolean as range): True, if the event affects the traffic flow in both directions of a two-way road, and False otherwise; and three object properties that represent relations between locations:
- **flowsTo** (with Location as domain and range): relates two locations, L and M , such that L is a two-way road and M indicates in which direction the traffic event affects the traffic lanes of L ;
- **isReferenceFor** (with Location as domain and range): relates two locations, M and L , such that M is a reference point for a traffic event that occurred in L ;
- **isRestrictedTo** (with Location as domain and range): relates two locations, L and M , such that L is restricted to the geographic area defined by M (usually M indicates a neighborhood or a city).

Time of a traffic event: A traffic event E , described in a tweet W , may have three different timestamps: (1) the timestamp when W was posted (indicated as an attribute of W); (2) the timestamp when the news about E was published (included in the text of W , usually as a clearly indicated heading, such as “[Monday, 10:30AM]”); (3) the timestamp when E occurred or will occur (often a noun phrase in the text of W , such as “early in the morning”, or a noun, such as “tomorrow”). To describe such timestamps, TEDO has a class:

- **Time:** a placeholder for the three types of timestamps of a traffic event. That is, an instance of Time has no specific semantics and is just an artifact to group the timestamps of a traffic event. As such, it could have been modeled as a *blank node* [4]; an object property:
- **hasTime** (with TrafficEvent as domain and Time as range): associates a traffic event with a single instance of Time; and three datatype properties, as explained above:
- **hasPostingTime** (with Time as domain and the type dateTime as range): the unique timestamp when the tweet was posted;
- **hasPublicationTime** (with Time as domain and the type dateTime as range): the unique timestamp when the news expressed in the tweet was published;
- **hasEventTime** (with Thing as domain and the type string as range): a unique noun phrase in the tweet indicating when the traffic event occurred or will occur.

Traffic intensity: The class TrafficSituation has a specific datatype property:

- **hasTrafficIntensity** (with TrafficSituation as domain and an enumeration type as range): describes the traffic intensity as “G”, for good traffic, “H”, for heavy traffic, and “S”, for slow or stopped traffic.

Causal relation between traffic events: To reflect that a traffic event may be the cause of another traffic event, TEDO includes an object property:

- **causes** (with TrafficEvent as domain and range): indicates that a source event causes a target event.

3. Tagged tweets, dependency trees and the representation of a tweet in TEDO

The interpretation of a tweet involves several tasks: (i) Tweet tokenization; (ii) Entity extraction; (iii) Geocoding; (iv) Relation extraction; and (v) RDF generation. Section 4 provides details about the algorithms that implement such tasks. This section defines and illustrates the use of the data structures that are output by the entity and the relation extraction tasks, which helps understanding how a tweet is represented in TEDO. Section 3.1 defines the tags used to mark the tweet text elements. Section 3.2 introduces the notion of *dependency tree*, which captures the relations between the tagged text elements of a tweet. Section 3.3 contains examples of tagged tweets and dependency trees. Finally, Section 3.5 illustrates how to represent a tweet in TEDO, based on the corresponding tagged tweet and dependency tree.

3.1. Tagged tweets

The entity extraction task is similar to the named entity recognition (NER) [12] process (also known as entity identification, entity chunking and entity extraction). It receives as input a tweet W and returns a tagged tweet T , that is, the tweet W with text elements classified with the help of tags. A *tagged text element* of T is a pair (e, t) , where e is a text element of T and t is the corresponding tag in T .

Table 4 summarizes the tags adopted and the corresponding TEDO terms. Note that there is a one-to-one correspondence between the tags in the second column and the TEDO terms in the third column, which is explored by the RDF generation task, illustrated in Section 3.5. The rest of this section lists the tags, grouped by role, and the most common text elements for each tag (the words are in Portuguese, followed by their translations), extracted from a testing corpus (see Section 4.2). We note that only the most significant text elements occurring in tweets are tagged, while all others are discarded.

Traffic event tags:

- **<interdiction>**: used to tag words such as “interdição” (“interdiction”) and “fechada” (“closed”);
- **<accident>**: used to tag words such as “colisão” (“collision”), “engavetamento” (“pile-up”), “choque” (“crash”), “capotamento” (“turn over”), “queda” (“fall”), “atropelamento” (“running over”);
- **<breakdown>**: used to tag words such as “enguiçado” (“breakdown”) and “pane” (“breakdown”);
- **<traffic>**: used to tag words such as “tráfego” (“traffic”) and “trânsito” (“traffic”);
- **<weather-condition>**: used to tag words and noun phrases such as “chuva” (“rain”), “alagamento” (“flooding”), “bolsão d’água” (“pocket of water”);
- **<other-event>**: used to tag less frequent words, such as “manifestação” (“manifestation”), which describe events that affect traffic.

Location tags:

- **<location-name>**: used to tag words that indicate the name of a location;
- **<both-direction>**: used to tag noun phrases, such as “nas duas direções” (“in both directions”) and “nos dois sentidos” (“in both directions”), which indicate that the event affects the flow of traffic in both directions;
- **<direction>**: used to tag words, such as “direção” (“direction”) and “sentido” (“direction”), which indicate in which direction the flow of traffic is affected by the event;

Table 4
Summary of tags and related vocabulary.

#	Tag	TEDO term	Examples of text elements tagged
1	<interdiction>	Interdiction	“interdicao” (“interdiction”), “fechada” (“closed”)
2	<accident>	Accident	“colisao” (“collision”), “engavetamento” (“pile-up”), “choque” (“crash”), “capotamento” (“turn over”), “queda” (“fall”), “atropelamento” (“running over”)
3	<breakdown>	Breakdown	“enguicado”, “pane” (“breakdown”)
4	<traffic>	TrafficSituation	“trafego” (“traffic”), “transito” (“traffic”)
5	<weather-condition>	WeatherCondition	“chuva” (“rain”), “alagamento” (“flooding”), “bolsao d agua” (“pocket of water”)
6	<other-event>	OtherEvent	“manifestacao” (“manifestation”)
7	<location-name>	hasLocationName	(names given by a gazetteer)
8	<both-directions>	affectsBothDirections	“nas duas direções” (“in both directions”) and “nos dois sentidos” (“in both directions”)
9	<direction>	flowsTo	“direcao”, “sentido” (“direction”)
10	<reference>	isReferenceFor	“altura” (“near”), “perto de”, “proximo de” (“close to”)
11	<restriction>	isRestrictedTo	“em” (“in”)
12	<co-reference>	sameAs	“no local” (“in the area”)
13	<event-time>	hasEventTime	“neste momento” (“at this moment”), “agora”, (“now”), “em 1 hora” (“in one hour”)
14	<actor>	hasActor	“carro” (“car”), “onibus” (“bus”), “motocicleta” (“motorcycle”)
15	<G>	“G”	“bom” (“good”), “livre” (“free”)
16	<H>	“H”	“intenso” (“intense”), “com retencoes” (“with retentions”)
17	<S>	“S”	“lento”, “lentidao” (“slow”), “parado” (“stopped”), “congestionamento” (“congestion”), “retencao” (“retention”)
18	<causes>	causes	“causou” (“caused”), “gerou” (“generated”), “ocasionou” (“resulted in”), “complica” (“complicates”)

- <reference>: used to tag words and noun phrases, such as “altura”, “perto de” (“near”) and “próximo de” (“close to”), which indicate a reference location for the event;
- <restriction>: used to tag words, such as “em” (“in”), which indicate a location that restricts the area affected by the event;
- <co-reference>: used to tag noun phrases, such as “no local” (“in the area”), which indicate that the location of a second event is the same as that of the first event.

Event time tag:

- <event-time>: used to tag words and noun phrases, such as “neste momento” (“at this moment”), “agora” (“now”) and “em 1 hora” (“in one hour”), which designate an event time.

Actor tag:

- <actor>: used to tag words, such as “carro” (“car”), “ônibus” (“bus”), “motocicleta” (“motorcycle”), which designate an actor of a traffic event.

Traffic intensity tags:

- <G>: used to tag words such as “bom” (“good”) and “livre” (“free”);
- <H>: used to tag words and noun phrases such as “intenso” (“intense”) and “com retenções” (“with retentions”);
- <S>: used to tag words such as “lento” (“slow”), “lentidão” (“slow”), “parado” (“stopped”), “congestionamento” (“congestion”), “retenção” (“retention”).

Causal tag:

- <causes>: used to tag words, such as “causou” (“caused”), “gerou” (“generated”), “ocasionou” (“resulted in”), “complica” (“complicates”), which designate the causal relation between traffic events.

3.2. Dependency trees

The relation extraction task refers to the problem of detecting relations between the entities expressed in the tagged tweet. It receives as input a tagged tweet T and returns a *dependency tree*

$\Theta_T = (N_T, E_T)$, where N_T is the set of tagged text elements of T and E_T is a set of arcs that indicate how the tagged text elements in N_T are related. We note that the relation extraction task actually computes a tree Θ_T , and not a generic graph, as further discussed in Section 4.1.2.

Each line of Table 5 indicates the domain and range of a possible relation between two classes of tagged text elements. For example, Line 1 indicates that a tagged text element K whose tag is any of the traffic event tags may be related to a tagged text element L whose tag is <actor>. Note that the correspondence between pairs of classes of tagged text elements and TEDO terms is somewhat more complex than in Table 4, a point that will be illustrated in Section 3.5.

3.3. Examples of tagged tweets and dependency trees

We illustrate the entity and relation extraction tasks with the help of tweets in Portuguese, taken from “@odia24horas” [13] (a translation is provided after the original text). As already mentioned, Section 4 provides additional details about the implementation of such tasks. At this point it suffices to observe that these tasks are implemented as classifiers, trained over a tweet corpus. In particular, we observe that the relation extraction task almost always identifies the first location name that appears in a tagged tweet as the primary location of the event and treats the other location names as secondary locations. This is an intuitive explanation of the observed behavior of the classifier that implements the relation extraction task, after the training step over a tweet corpus, and probably reflects the fact that the terms (such as “direction”) used to relate the primary location to a secondary location in a tweet always refer to the primary location before the secondary location. Likewise, the relation extraction task almost always identifies the first event that appears in a tweet as a cause for a second event, if present in the tagged tweet. Again, this is an intuitive explanation of the observed behavior of the relation extraction task and probably reflects the fact that the terms (such as “cause”) used to report an event as a cause for a second event in a tweet always refer to the causing event before the caused event.

Example 1.

Tweet: “Acidente entre 2 carros na Av das Américas na pista sentido Grota Funda próximo ao número 19880”

Table 5
Summary of the relations captured in a tweet dependency graph.

#	Domain tag class	Range tag class	TEDO term
1	(Any of the traffic event tags)	<actor>	hasActor
2	(Any of the traffic event tags)	<event-time>	hasTime + hasEventTime
3	(Any of the traffic event tags)	<causes>	(inverse of causes)
4	<causes>	(Any of the traffic event tags)	causes
5	(Any of the traffic event tags)	(Any of the traffic event tags)	causes
6	(Any of the traffic event tags)	<location-name> or <co-reference>	hasPrimaryLocation + hasLocationName
7	<location-name> or <co-reference>	<both-directions>	affectsBothDirections
8	<location-name> or <co-reference>	<direction>	(inverse of flowsTo)
9	<direction>	<location-name>	flowsTo
10	<location-name> or <co-reference>	<reference>	isReferenceFor
11	<reference>	<location-name>	(inverse of isReferenceFor)
12	<location-name> or <co-reference>	<restriction>	(inverse of isRestrictedTo)
13	<restriction>	<location-name>	isRestrictedTo
14	<traffic>	<G>	hasTrafficIntensity
15	<traffic>	<H>	hasTrafficIntensity
16	<traffic>	<S>	hasTrafficIntensity

Translation: “Accident between 2 cars at Av das Américas in lane direction Grota Funda near number 19880”

Date: 05/03/2012 07:07:01 (the timestamp the tweet was sent)

The entity extraction task generates the tagged tweet shown in Fig. 2 from the tweet text. Intuitively, the entity extraction task identifies an accident (“Acidente”), with one actor (“2 carros” – not treated as two separated actors in this case), and three location names (“Av. das Américas”, “Grota Funda” and “número 19880”). It also identifies the direction of the traffic flow (“sentido”) that the accident affects and a reference (“próximo a”) for the accident.

The relation extraction task then creates the dependency tree shown in Fig. 3 from the tagged tweet. The numbers labelling the arcs refer to the lines in Table 5; they are used for explanation purpose only and are not part of the definition of the dependency tree. The arc labeled with “1” indicates that the relation extraction task relates N_2 to N_1 as per Line 1 of Table 5. As explained before, the relation extraction task almost always considers the first location that appears in the tweet as the primary location and treats the other locations as secondary locations. Hence, the arc labeled with “6” indicates that the relation extraction task relates N_2 to N_3 as per Line 6 of Table 5. But the task relates N_2 neither to N_6 nor to N_7 . Rather, it relates N_3 to N_4 and N_4 to N_6 as per Lines 8 and 9. Likewise, it relates N_3 to N_5 and N_5 to N_7 as per Lines 10 and 11.

Example 2.

Tweet: “Interdição AvBrasil em Manguinhos complica trânsito na LinhaAmarela”

Translation: “Interdiction of AvBrasil in Manguinhos complicates traffic in LinhaAmarela”

Date: 24/03/2012 18:37:23 (the timestamp the tweet was sent)

The entity extraction task generates the tagged tweet shown in Fig. 4 from the tweet text. Intuitively, the entity extraction task identifies two traffic events (“interdição” and “trânsito”), three location names (“Av. Brasil”, “Manguinhos” and “Linha Amarela”), a causal text element (“complica”) and a restriction (“em”).

The relation extraction task then creates the dependency tree shown in Fig. 5 from the tagged tweet. The arcs labeled with “3”

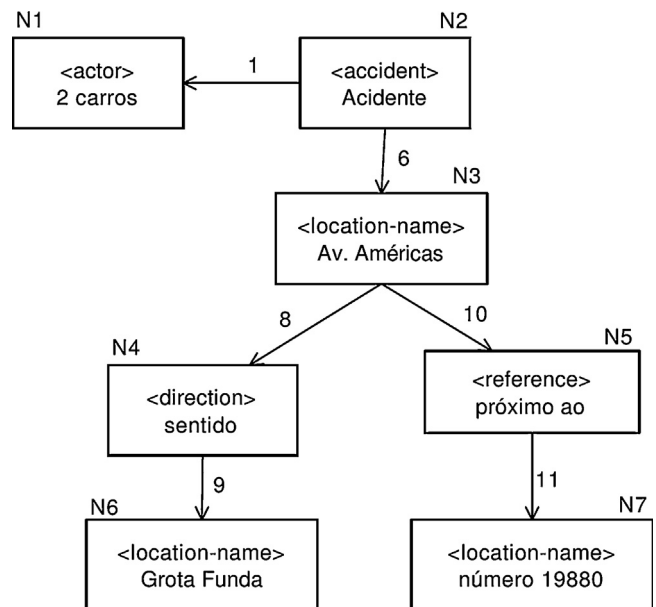


Fig. 3. The tweet dependency tree for the tweet of Example 1.

and “4” indicate that the relation extraction task relates N_1 to N_2 and N_2 to N_3 as per Lines 3 and 4 of Table 5, based on the causal text element. These two arcs indicate that the first traffic event is a cause for the second event. Indeed, as explained before, the relation extraction task almost always considers the first event to be a cause for the other events that appear in the tweet text. The other arcs of the dependency tree in Fig. 5 have explanations similar to those of the dependency tree of Fig. 3.

Example 3.

Tweet: “Queda de motociclista na pista central da AvBrasil, altura de Bonsucesso sentido Centro trânsito é lento no local”

Translation: “Motorcycle fall in the central lane of AvBrasil, near Bonsucesso direction Centro traffic is slow in the area”

Date: 26/03/2012 07:19:27 (the timestamp the tweet was sent)

Acidente	entre	2 carros	na	Av. das Américas	na pista	sentido	Grota Funda	próximo ao	número 19880
<accident>		<actor>		<location-name>		<direction>	<location-name>	<reference>	<location-name>

Fig. 2. Entities of Example 1 tweet.

<i>Interdição</i>	<i>#AvBrasil</i>	<i>em</i>	<i>Manguinhos</i>	<i>complica</i>	<i>trânsito</i>	<i>na</i>	<i>#LinhaAmarela</i>
<interdiction>	<location-name>	<restriction>	<location-name>	<causes>	<traffic>		<location-name>

Fig. 4. Entities of Example 2 tweet.

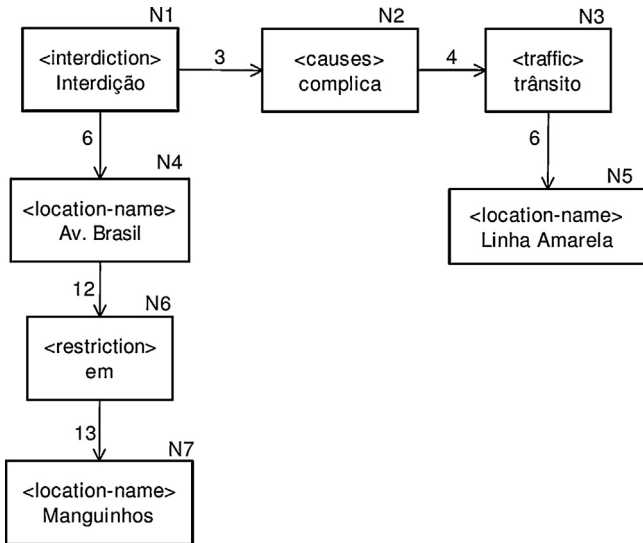


Fig. 5. The tweet dependency tree for the tweet of Example 2.

The entity extraction task generates the tagged tweet shown in Fig. 6 from the tweet text. Intuitively, the entity extraction task identifies two traffic events (“queda” and “trânsito”), an actor (“motocicleta”), three location names (“Av Brasil”, “Bonsucesso”

and “Centro”). It also identifies a restriction (“altura de”), a direction of the flow (“sentido”), a traffic intensity (“lento”) and a co-reference (“no local”).

The relation extraction task then creates the dependency tree shown in Fig. 7 from the tagged tweet. The arc labeled with “5” tells that the relation extraction task relates N_2 to N_3 , as per Line 5 of Table 5, to indicate that the first traffic event is a cause for the second event. The arc labeled with “6” from N_3 to N_6 indicates that the relation extraction task relates these two nodes as per Line 6 of Table 5. Node N_6 acts as a co-reference for the primary location of the first event. The other arcs of the dependency tree in Fig. 7 have explanations similar to those of the dependency tree of Fig. 5.

3.4. Geocoding

The geocoding sub-task determines the coordinates of the (named) locations described in a tweet, and indicates if the event occurred in a two-way street, and in which direction.

This sub-task uses the tagged tweet as a parameter of the SmartGeocode Algorithm [14]. This algorithm was built to georeference the location of events, using geocoding and routing services, from spatial descriptions commonly found in human communication. The output of SmartGeocode is added to the tagged tweet three tagged with < both-directions> and <location-coordinates>, whose parent is either <primary-location> and <secondary-location>.

<i>Queda de</i>	<i>motocicleta</i>	<i>na pista central da</i>	<i>Av. Brasil</i>	<i>altura de</i>	<i>Bonsucesso</i>	<i>sentido</i>	<i>Centro</i>
<accident>	<actor>		<location-name>	<restriction>	<location-name>	<direction>	<location-name>

<i>trânsito</i>	<i>é lento</i>	<i>no local</i>
<traffic>	<S>	<co-reference>

Fig. 6. Entities of Example 4 tweet.

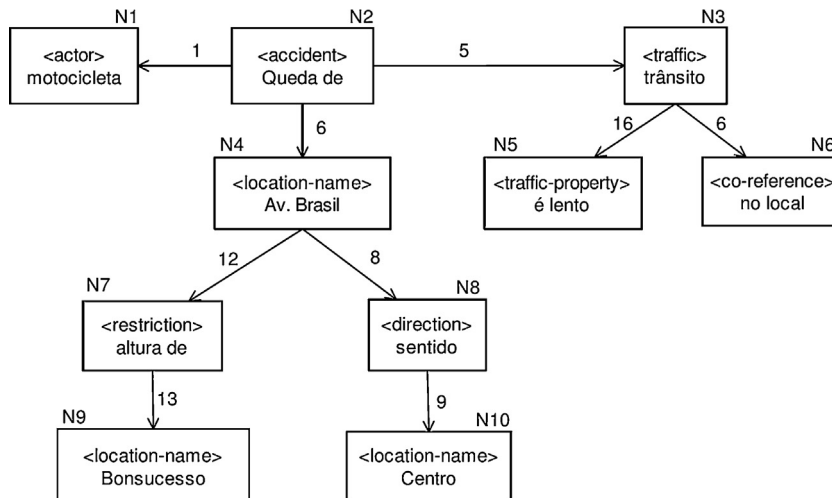


Fig. 7. The tweet dependency tree for the tweet of Example 4.

```

1. <tedo:Accident
      rdf:about='http://www.ld.inf.puc-rio.br/tedo/trafficevent/100''>
2.   <tedo:hasActor>2 carros</tedo:hasActor >
3.   <tedo:hasPrimaryLocation
      id="http://www.ld.inf.puc-rio.br/tedo/location/101">
4.   <tedo:hasSecondaryLocation
      id="http://www.ld.inf.puc-rio.br/tedo/location/102">
5.   <tedo:hasSecondaryLocation
      id="http://www.ld.inf.puc-rio.br/tedo/location/103">
6.   <tedo:hasTime id="http://www.ld.inf.puc-rio.br/time/104">
7. </tedo:Accident>
8. <!-- Primary Location -->
9. <tedo:Location
      rdf:about="http://www.ld.inf.puc-rio.br/location/101">
10.  <tedo:hasLocationName>Av. Das Amricas</tedo:hasLocationName>
11.  <tedo:hasCoordinates>-22.998864, -43.365984</tedo:hasCoordinates>
12.  <tedo:flowsTo
      rdf:resource="http://www.ld.inf.puc-rio.br/location/102">
13.  <tedo:isReferenceFor
      rdf:resource="http://www.ld.inf.puc-rio.br/location/103">
14. </tedo:Location>
15. <!-- Secondary Location -->
16. <tedo:Location
      rdf:about="http://www.ld.inf.puc-rio.br/location/102">
17.  <tedo:hasLocationName>Grota Funda</tedo:hasLocationName>
18.  <tedo:hasCoordinates>-23.015379, -43.521634</tedo:hasCoordinates>
19. </tedo:Location> \
20. <!-- Secondary Location -->\
21. <tedo:Location
      rdf:about="http://www.ld.inf.puc-rio.br/location/103">
22.  <tedo:hasLocationName>Nmero 19880</tedo:hasLocationName>
23.  <tedo:hasCoordinates> -23.016279, -43.514426</tedo:hasCoordinates>
24. </tedo:Location>
25. <!-- Time -->
26. <tedo:Time
      rdf:about="http://www.ld.inf.puc-rio.br/time/104">
27.  <tedo:hasPublicationTime>05/03/2012 07:07:01</tedo:hasPublicationTime>
28. </tedo:Time>

```

Fig. 8. RDF triples of the tweet in Example 1.

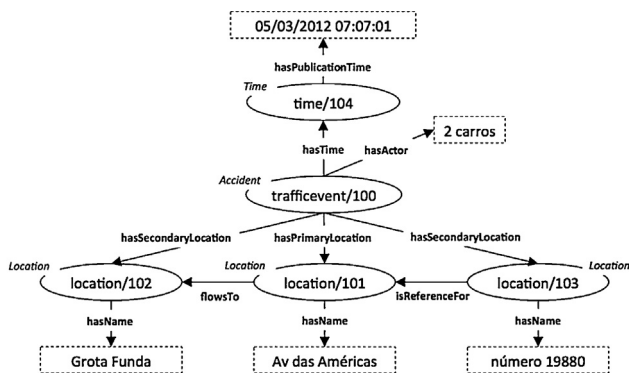
3.5. RDF generation

The RDF generation task translates a tagged tweet T and the corresponding dependency tree Θ_T into a set R_T of RDF triples, constructed using the TEDO vocabulary. The translation is straightforward and will only be illustrated with an example.

Example 4. Using fictional IRIs, the RDF generation task translates the tagged tweet and the dependency tree in Example 1 of Section 3.3 to the triples in RDF/XML [15] shown in Fig. 8 (see also Fig. 9 for the corresponding RDF graph).

In Fig. 8, each line indicates:

- Line 1: a traffic event of the class Accident;
- Line 2: an actor (“2 carros”) of the traffic event;
- Line 3: the primary location of the traffic event;
- Lines 4 and 5: secondary locations of the traffic event;
- Line 6: the time of the traffic event;
- Lines 10 and 11: the values of properties hasLocationName and hasCoordinates of the primary location;



Tweet: "Acidente entre 2 carros na Av das Américas na pista sentido Grota Funda próximo ao número 19880"
Translation: "Accident between 2 cars at Av das Américas in lane direction Grota Funda near number 19880"
Date: 05/03/2012 07:07:01

Fig. 9. RDF graph corresponding to the triples defined in Example 5.

Line 12: the secondary location, named “Grota Funda”, defines the flow of direction affected;
 Line 13: the secondary location, named “Número 19880”, indicates a reference for the accident;
 Lines 17 and 18: the values of properties hasLocationName and hasCoordinates of the first secondary location;
 Lines 22 and 23: the values of properties hasLocationName and hasCoordinates of the first secondary location;
 Line 27: the publication time of the tweet.

4. Implementation and experiments

This section first describes some implementation issues of the entity extraction and the relation extraction subtasks and then presents the results of the experiments carried out with a corpus of real-world traffic tweets.

4.1. Tweet interpretation

4.1.1. Attributes used for entity extraction

The entity extraction subtask was implemented as a classification process, adopting a machine learning approach. The objective is to classify each text element of a tweet to a tag, as described in Table 4. We first tokenize the tweet using the Web service called F-EXT [16], which collects all the tokens from a sentence with their morphosyntactic characteristics.

The names of these characteristics, followed by the corresponding variable within parenthesis, used are listed below.

- **Token** (W_i) – the content of token X_i .
- **Simple Token** (SMW_i) – the simplified content of token X_i (in lower case, without any special characters and punctuation).
- **Simplified Token** (SW_i) – the simplified content of token X_i (in lower case, without any special characters, and removing letters, punctuation or numbers of size 1).
- **Part-of-Speech** (POS_i) – the part of speech of token X_i .
- **Stemmed Word** (STW_i) – the root of the word present in token X_i . For example, the root of “blocked” is “block”.

These characteristics are used to define the features that corresponds to the input variables of the classification method, and they are:

- **CurrT**(X) – current token X_i .
- **PrevT**(X, N) – the N th token before the current token X . (Usually, we use $N = 1$ and $N = 2$.)
- **NextT**(X, N) – the N th token after the current token X . (Usually, we use $N = 1$ and $N = 2$.)
- **CurrWSC** – indicates whether X_i begins in upper-case and does not have any other upper-case letter.
- **LocType** – indicates whether the lower-case X_i represents a designation, that is, $X_i \in \{av, ave, avenue, avenues, hwy, highway, r, rd, road, roads\}$.

For the classification process, we used the SMO implementation [17] of the SVM Family (available on Weka [18] version 3.6.5). Among the several classifier that we tested, the SVM shows the best results.

Example 5.

Tweet: “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”

Let $i = 3$: (that is, the token “Clemente” of the text in Portuguese). Then, the input features for the 3rd token of the tweet that are given to the classifier correspond to:

- **CurrT**(W_i): Clemente
- **CurrT**(SW_i): clemente
- **CurrT**(POS_i): NAME
- **PrevT**($W_i, 2$): Rua
- **PrevT**($POS_i, 2$): NAME
- **PrevT**($W_i, 1$): São
- **PrevT**($POS_i, 1$): NAME
- **NextT**($W_i, 1$);
- **NextT**($POS_i, 1$);
- **NextT**($W_i, 2$): em
- **NextT**($POS_i, 2$): PREPOSITION
- **LocType**(i): NO
- **CurrWSC**(i): YES

For this example, the classifier output the tag <location-name>.

4.1.2. Attributes used for relation extraction

The relation extraction subtask computes the tweet dependency tree G_T of a tagged tweet T . To compute G_T , each pair of tagged text elements, K and L , is assigned a set of attributes, which is used by a machine learning algorithm to learn about the nature of the (possible) relations between K and L . The implementation first uses the large margin structured perceptron [19] to compute a weight for each edge. This algorithm builds a directed graph with weighted edges; then it finds the Maximum Spanning Tree of the directed graph to “discover” the relations.

The attributes of a node K of G_T are:

- **Word** (W) – indicates the words of the text element in K .
- **Simplified Word** (SW) – the simplified words of the text element in K .
- **Ruler Entity** (RE) – the named entity of K (if it is not a Relevant Entity, POS is used).
- **Named Entity** (NE) – the named entity of K (ignored if it is not a Relevant Entity).
- **Punctuation** ($PUNCT$) – indicates whether there is punctuation in the text element of K .

The features used for relation extraction are:

- **PossRel** – indicates whether K and L may have a relation (as in Table 5).
- **ConcT** (Y) – represents the concatenated words of the text elements Y_i and Y_j respectively of K and L .
- **BeT** (Y) – represents all tokens obtained from the text elements Y_i and Y_j respectively of K and L .
- **NearT** (Y, N) – represents all tokens in $Y_{i-1}, \dots, Y_{i-N}, Y_{j-1}, \dots, Y_{jN}, Y_{i+1}, \dots, Y_{i+N}, Y_{j+1}, \dots, Y_{j+N}$. It counts as a single attribute, separated by character “_”.
- **AbsLocPair** – indicates whether K and L have a relation, where the tag of K is <restriction> and that of L is <location-name>; otherwise, no value is used.
- **MetaWithLoc** – indicates whether K and L have a relation, where the tag of K is <reference> or <direction> and that of L is <location-name>; otherwise, no value is used.

Example 6. Consider the following tweet, with a pair of entities, Y_i and Y_j , tagged with <location-name> (“Rua São Clemente”) and <G> (“bom”).

Tweet: “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.”

Let $i = 4$ and $j = 1$.

- **PossRel:** PR_YES
- **Conc** (NE): <G>_<location-name>
- **ConcT** (SW): trânsito bom_Rua São Clemente
- **BetT** (RE): <restriction> <location-name>, PREPOSITION
- **BetT** (W): em Botafogo, com
- **NearT** (RE, 3): <restriction>_<location-name>PREPOSITION_, PREPOSITION_<reference>PREPOSITION
- **NearT** (SW, 2): com__ em_a __ em_
- **NearT** (NE, 2): <location-name>_<restriction>_<reference>_<location-name>_
- **AbsLocPair:** _<restriction>_<location-name>_

4.2. Results

We selected 690 traffic-related tweets, extracted from “@odi-a24horas” [13] and “@operacoesrio” [20] sampled in a period extended from March 1st, 2013 to April 30th, 2013. We manually constructed a corpus with the corresponding tagged tweets. 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.

Recall that the accuracy, recall, precision and F-measure of a classification task are defined as follows:

$$\text{Precision} = \frac{T_p}{(T_p + F_p)}$$

$$\text{Recall} = \frac{T_p}{(T_p + F_n)}$$

$$\text{Accuracy} = \frac{(T_p + T_n)}{(T_p + T_n + F_p + F_n)}$$

$$F\text{-measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

where the variables T_p , F_p , T_n and F_n are defined as in Table 6.

We applied a ten-fold cross-validation procedure to evaluate the performance of the entity extraction sub-task. Recall that this procedure divides the corpus in 10 equally sized parts and performs 10 tests. In each test, one part is separated for the performance evaluation and the remaining 9 parts are used for training. Table 7 shows the results of the performance measures using the ten-fold cross-validation procedure. Note that the proposed tagging method had accuracy higher than 70% in the majority of the cases.

Finally, the performance of the relation extraction sub-task was evaluated on a corpus of 200 tagged tweets, manually annotated according to the TEDO ontology. We obtained a mean accuracy of 73%, with a standard deviation of 13.82%.

Table 6
Result classification used on the performance measures.

Variable	Result	Annotation	Classification
T_p	Positive	Positive	True positive
F_p	Positive	Negative	False positive
T_n	Negative	Positive	False negative
F_n	Negative	Negative	True negative

Table 7

Performance measures of the tagging task using a ten-fold cross-validation procedure.

Tag	Accuracy	Precision	Recall	F-measure	F-measure Std. deviation
<accident>	95.25	98.66	96.16	97.37	4.74
<actor>	79.10	93.94	82.01	86.41	15.79
<both-directions>	92.18	93.75	95.00	94.31	15.03
<breakdown>	95.23	97.77	96.29	96.96	8.50
<causes>	82.93	94.41	86.34	89.75	10.40
<co-reference>	97.28	100.00	97.28	98.53	3.03
<direction>	93.68	99.76	93.82	96.53	4.74
<event-time>	56.99	62.28	62.70	61.80	35.79
<interdiction>	47.09	62.61	51.50	54.55	28.93
<location-name>	87.53	94.49	92.00	93.21	3.87
<other-event>	49.60	75.99	54.27	60.44	28.01
<reference>	93.91	99.14	94.66	96.78	2.88
<restriction>	91.87	100.00	91.87	95.02	9.70
<traffic>	90.84	96.98	93.22	95.00	4.62
<weather-condition>	75.76	84.72	81.19	82.22	23.38
<G>	75.46	89.35	80.11	84.01	15.89
<H>	87.95	96.54	90.00	93.07	7.82
<S>	88.94	95.09	90.53	92.59	14.78

5. Related work

Proactive behavior provided the basic motivation for the automated traffic-related tweet interpretation tool described in this paper. Proactive computing is investigated, for example, in [1,21,22].

The development of the TEDO domain ontology adopted concepts from several references. Events, according to Sowa [5], describe discrete changes that occur in processes. Kaneiwa et al. [6] distinguished between events and objects and considered that events have instances. Kaneiwa et al. [6] also defined several types of relations between events: causal, temporal or spatial. Worboys and Hornsby [7] introduced geospatial and spatiotemporal relations between events. TEDO borrows the notion of event from these references and models traffic as a process characterized by inter-related discrete events.

The literature offers examples of domain ontologies specifically designed for traffic-related applications. Yue et al. [8] defined an ontology to model traffic accidents to facilitate interoperability between traffic management systems. Wang and Wang [9] introduced the TADO ontology (Traffic Accident Domain Ontology) to enable semantic search over traffic accident databases. TEDO borrowed some concepts from these two traffic accident ontologies, specifically we extended traffic accidents modeled in TADO with other types of traffic events.

The automated interpretation of tweets has been extensively covered in the literature. In what follows, we focus on a few references that are closely related to our traffic-related interpretation tasks, starting with the problem of extracting and geocoding the location of a traffic event reported in a tweet.

Sakaki et al. [23] analyzed tweets with respect to the relevance of their content and spatiotemporal information. The analysis of the spatial aspect considered only the registered location (the position of the GPS enabled device when the tweet was sent). The authors explored, as use cases, the problems of estimating the location of an earthquake center and the trajectory of a typhoon. The authors supposed that the registered location was near to the location of the reported event.

Earle et al. [24] introduced a model for earthquake detection. Event detection was based on the increasing rate of earthquakes tweets. Tests were performed over downloaded tweets containing the words earthquake, and its translation in other languages, that were collected during four months. The event location was inferred either from the registered location of the tweet or by using the

Google Maps Geocoding Service to geocode the location name stored in the user profile. However, because the geocoding of location names is not as precise as for addresses, the use of the location name stored in the user profile may not lead to accurate results.

MacEachren et al. [25] handled a missing registered location by considering the location name stored in the user profile, a place reference extracted from the tweet content or a hashtag associated with a place. Predefined keywords and phrases were selected to refine queries on tweets. The resulting tweets were then analyzed and the extracted locations were geocoded using GeoNames. As an example of an application, the authors analyzed tweets to outline the role of one church to organize relief efforts to help victims of the Haitian earthquake.

Chen et al. [26] adopted a rule-based tagger to extract potential place names from a given text and to match the extracted names against a location database. Borges et al. [27] used predefined patterns to extract addresses from Web pages using a set of regular expressions.

The implementation described in Section 4 adopts a strategy similar to that of Ref. [25] to extract and geocode the location of a traffic event directly from the tweet text. Indeed, unlike the applications covered in Refs. [23,24], neither the registered location of the tweet nor the location name stored in the user profile proved to be useful to infer the location of a traffic accident in the traffic-related tweet corpus we analyzed. Using a set of regular expressions, as in Refs. [26,27], to extract the location of a traffic-related event from the tweet text also proved not to be effective in the traffic-related tweet corpus we considered.

The tagging described in Sections 3.1 and 4.1.1 is closely related to the familiar named entity recognition (NER) task [12]. Ritter et al. [28] proposed an entity recognition approach for tweets, calling attention to idioms used in the Internet. Jung [29] also used tweets as a case study. As a reference close to the problem addressed in this paper, we may quote Carvalho et al. [30], which addressed the problem of classifying tweets that indicate the occurrence of traffic-related events, using machine-learning techniques over a dataset with traffic-related and non-related messages.

The tweet structuring described in Sections 3.2 and 4.1.2 is in turn similar to the relation extraction problem. The most successful approaches to address this problem apply supervised Machine Learning to construct classifiers using features extracted from hand-labeled sentences of a training corpus [31–34]. To address the scalability problem in relation extraction frameworks, weak supervision methods were introduced, based on the idea of using a database with structured data to heuristically label a text corpus. For example, Bellare and McCallum [35] used BibTex records to train a CRF extractor for 12 bibliographic relations. Wu and Weld [36] used weak supervision to learn relations from the articles, using Wikipedia infoboxes as a database of relation instances. Wu and Weld [37] extended this strategy by using smoothing over an automatically generated infobox taxonomy. Mintz et al. [38] coined the term *distant supervision* to replace the term weak supervision. They applied Freebase facts to create relation extractors from Wikipedia, achieving an average precision of approximately 67.6% for the top 100 relations.

The tool described in Section 4 adopts Machine Learning techniques to implement the entity and relation extraction tasks. Although structuring raw text data and extracting relevant information is a non trivial task, as already observed, tweets that describe traffic-related events and that are distributed by government agencies or by news agencies use a much more regular and well-behaved prose than generic user-generated tweets. This characteristic facilitated automating their interpretation and achieving high precision and recall. However, as discussed

in Sections 4.1.1 and 4.1.2, the high precision and recall were only obtained by: (1) the selection, after extensive testing, of adequate classification algorithms (the SMO implementation [17] of the SVM Family, available on Weka [18] version 3.6.5, for the entity extraction task, and the large margin structure perceptron, for the relation extraction task [19]); (2) a careful attribute engineering; and (3) an adequate training of the classification algorithms, using a supervised approach and an appropriate hand-labelled tweet corpus. Also, the extraction of the tokens from each tweet text and the definition of the morphosyntactic characteristic of each token benefited from the use of the F-EXT Web service [16].

Finally, the RDF generation task proved to be far simpler than the generic problem of translating natural language to RDF. Indeed, we do not directly translate the tweet text to RDF, but rather we first transform the tweet into a tagged tweet and a dependency tree, which then almost directly induce the set of RDF triples that represent the tweet, as the example in Section 3.5 illustrates.

6. Conclusions

In this paper, we concentrated on the task of interpreting traffic-related tweets distributed by traffic authorities and news agencies. The result of the interpretation of a tweet is expressed as a set of RDF triples, which uses a specific domain ontology, called TEDO, that models traffic-related situations as events, composed of actors, locations and timestamps. The main contribution of the paper is an automatic tweet interpretation tool, based on Machine Learning techniques, that achieves good performance for traffic-related tweets. The tool explores the fact that such tweets have a fairly regular and simple syntactical structure, not observed in user-generated tweets. In particular, given a traffic-related tweet, the tool uses named-entity recognition techniques to identify the location of the event the tweet describes and relation extraction methods to capture relations between the components of the event. Finally, we described in detail experiments with real traffic-related tweets, which indicate that the tool achieves high precision and recall.

The motivation for the work reported in this paper was a prototype application designed to monitor truck fleets, which includes the tool. The application has been deployed to monitor a medium-size truck fleet (about 500 trucks), operated by a liquid gas distribution company, and a large truck fleet (about 5000 trucks), operated by a fuel distribution company. In addition to cost reduction and better fleet management, the application helped improve customer satisfaction by increasing the accuracy of the predicted time-of-delivery, a crucial aspect both for the customers of the liquid gas distribution company and for the gas station owners, served by the fuel distribution company.

In future developments, we plan to include a minimum classification for the actors involved in a traffic event to try to infer the severity and therefore the duration of the event. For example, accidents involving victims, typically pedestrians and motorcycle drivers, or multiple vehicles, have a higher severity. In addition, we plan to extend the tool reported in this paper to languages other than Portuguese, since we have strong reasons to believe that the proposed methodology should work for different languages, since it is ultimately based on a POS tagger that proved to be efficient for a variety of languages [16].

References

- [1] D. Tennenhouse, *Proactive computing*, *Commun. ACM* 43 (2000) 43–50.
- [2] geoRSS, 2014 URL: <http://georss.org>.
- [3] Ogc, Open GEOSMS Standard – Core, 2012 URL: <http://www.opengeospatial.org/standards/opengeosms>.
- [4] RDF 1.1 primer, w3c Working Group Note 24, 2014 URL: <http://www.w3.org/TR/2014/NOTE-rdf11-primer-20140624/>.
- [5] J.F. Sowa, *Ontology*, 2010 URL: <http://www.jfsowa.com/ontology/>.

[6] K. Kaneiwa, M. Iwazume, K. Fukuda, An upper ontology for event classifications and relations, in: M. Orgun, J. Thornton (Eds.), *AI 2007: Advances in Artificial Intelligence*, vol. 4830 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2007, pp. 394–403.

[7] M. Worboys, K. Hornsby, From objects to events: GEM, the geospatial event model, in: M. Egenhofer, C. Freksa, H. Miller (Eds.), *Geographic Information Science*, vol. 3234 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2004, pp. 327–343.

[8] D. Yue, S. Wang, A. Zhao, Traffic accidents knowledge management based on ontology, in: Proceedings of the 6th International Conference on Fuzzy Systems and Knowledge Discovery, vol. 7, FSKD'09, IEEE Press, Piscataway, NJ, USA, 2009, pp. 447–449, <http://dl.acm.org/citation.cfm?id=1802134.1802233>.

[9] J. Wang, X. Wang, An ontology-based traffic accident risk mapping framework, in: D. Pfoser, Y. Tao, K. Mouratidis, M. Nascimento, M. Mokbel, S. Shekhar, Y. Huang (Eds.), *Advances in Spatial and Temporal Databases*, vol. 6849 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2011, pp. 21–38.

[10] XML Schema Part 0: Primer, w3c Recommendation 2 May 2001, second ed., 28 October 2004, 2004 URL: <http://www.w3.org/TR/xmlschema-0/>.

[11] A. Borgida, R.J. Brachman, Conceptual modeling with description logics, in: *Description Logic Handbook*, 2003, 349–372.

[12] D. Nadeau, S. Sekine, A survey of named entity recognition and classification, *Lingvist. Investig.* 30 (2007) 3–26.

[13] Twitter Channel, O dia 24h, 2014 URL: <https://twitter.com/odia24horas>.

[14] F.d.C. Albuquerque, M.A. Casanova, J.A.F.d. Macedo, M.T.M.d. Carvalho, C. Renso, A proactive application to monitor truck fleets, in: *IEEE 14th International Conference on Mobile Data Management (MDM)*, 2013, vol. 1, IEEE, 2013, pp. 301–304.

[15] RDF 1.1 XML Syntax, w3c Recommendation, 25 February 2014, 2004 URL: <http://www.w3.org/TR/2014/REC-rdf-syntax-grammar-20140225/>.

[16] E. Motta, E. Fernandes, R. Milidiú, F-EXT-2.0: a web service for natural language processing, in: *PROPOR*, 2010, 27–30.

[17] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: *Advances in Kernel Methods*, MIT Press, 1999, pp. 185–208.

[18] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The Weka data mining software: an update, *ACM SIGKDD Explor. Newslett.* 11 (2009) 10–18.

[19] E.R. Fernandes, C.N. dos Santos, R.L. Milidiú, Latent structure perceptron with feature induction for unrestricted coreference resolution, in: *Joint Conference on EMNLP and CoNLL – Shared Task, CoNLL'12*, Stroudsburg, PA, USA, Association for Computational Linguistics, 2012, pp. 41–48.

[20] Twitter Channel, Operações rio, 2014 URL: <https://twitter.com/operacoesrio>.

[21] B. Magoutas, G. Mentzas, D. Apostolou, Proactive situation management in the future Internet: the case of the smart power grid, in: *2011 22nd International Workshop on Database and Expert Systems Applications (DEXA)*, IEEE, 2011, pp. 267–271.

[22] M.Y. Santos, A. Moreira, Guess: On the Prediction of Mobile, Movement-aware Applications for Sustainable Mobility: Technologies and Approaches, 2010, p. 87.

[23] T. Sakami, M. Okazaki, Y. Matsuo, Earthquake shakes twitter users: real-time event detection by social sensors, in: *Proceedings of the 19th International Conference on World Wide Web, ACM*, 2010, pp. 851–860.

[24] P.S. Earle, D.C. Bowden, M. Guy, Twitter earthquake detection: earthquake monitoring in a social world, *Ann. Geophys.* 54 (2012).

[25] A.M. MacEachren, A.C. Robinson, A. Jaiswal, S. Pezanowski, A. Saveljev, J. Blanford, P. Mitra, Geo-twitter analytics: applications in crisis management, in: *25th International Cartographic Conference*, 2011, 3–8.

[26] Y.-F.R. Chen, G. Di Fabbriozio, D. Gibbon, S. Jora, B. Renger, B. Wei, Geotracker: geospatial and temporal RSS navigation, in: *Proceedings of the 16th International Conference on World Wide Web, ACM*, 2007, pp. 41–50.

[27] K.A. Borges, A.H. Laender, C.B. Medeiros, C.A. Davis Jr., Discovering geographic locations in web pages using urban addresses, in: *Proceedings of the 4th ACM Workshop on Geographical Information Retrieval, ACM*, 2007, pp. 31–36.

[28] A. Ritter, S. Clark, O. Etzioni, et al., Named entity recognition in tweets: an experimental study, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics*, 2011, pp. 1524–1534.

[29] J.J. Jung, Online named entity recognition method for microtexts in social networking services: a case study of Twitter, *Expert Syst. Appl.* 39 (2012) 8066–8070.

[30] S. Carvalho, L. Sarmento, R. Rossetti, Real-time sensing of traffic information in twitter messages, in: *Proceedings of 4th Workshop on Artificial Transportation Systems and Simulation, IEEE*, 2010.

[31] J.R. Curran, S. Clark, Language independent NER using a maximum entropy tagger, in: *Proc. 7th Conf. on Natural Language Learning at HLT-NAACL 2003*, vol. 4, Association for Computational Linguistics, Stroudsburg, PA, USA, 2003, pp. 164–167, <http://dx.doi.org/10.3115/1119176.1119200>.

[32] J.R. Finkel, T. Grenager, C. Manning, Incorporating non-local information into information extraction systems by Gibbs sampling, in: *Proc. 43rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics*, Stroudsburg, PA, USA, 2005, pp. 363–370, <http://dx.doi.org/10.3115/1219840.1219885>.

[33] T.D. Nguyen, M. yen Kan, Keyphrase extraction in scientific publications, in: *Proc. Int'l. Conf. on Asian Digital Libraries, Springer*, 2007, pp. 317–326.

[34] G. Zhou, J. Su, Named entity recognition using an HMM-based chunk tagger, in: *Proc. 40th Annual Meeting on Association for Computational Linguistics*,

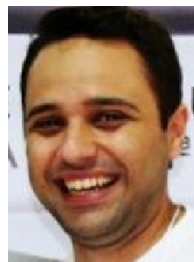
Association for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 473–480, <http://dx.doi.org/10.3115/1073083.1073163>.

[35] K. Bellare, A. McCallum, Learning extractors from unlabeled text using relevant databases, in: *Sixth International Workshop on Information Integration on the Web, 2007*.

[36] F. Wu, D.S. Weld, Autonomously semantifying Wikipedia, in: *Proc. 16th ACM Conf. on Information and Knowledge Management, ACM*, New York, NY, USA, 2007, pp. 41–50, <http://dx.doi.org/10.1145/1321440.1321449>.

[37] F. Wu, D.S. Weld, Automatically refining the Wikipedia infobox ontology, in: *Proceedings of the 17th International Conference on World Wide Web, ACM*, 2008, pp. 635–644.

[38] M. Mintz, S. Bills, R. Snow, D. Jurafsky, Distant supervision for relation extraction without labeled data, in: *Proc. Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, vol. 2, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 1003–1011.



Fabio Albuquerque has a Master degree in Informatics (2012) at PUC-Rio and a Bachelor degree in Information Technology (2007) at Centro Universitário da Cidade. He is a co-founder and technical lead at the Hivesoft company.



Marco A. Casanova graduated in Electronic Engineering from the Military Institute of Engineering in 1974, and obtained his PhD in Applied Mathematics from Harvard University in 1979. In 1998, he joined the Department of Informatics of the Pontifical Catholic University of Rio de Janeiro – PUC-Rio, where he is Associate Professor. In May 2012, he was appointed Coordinator of the Central Planning and Evaluation Office of PUC-Rio. His research interests concentrate on database conceptual modeling and construction of database management systems. For his contributions, in July 2012, he received the Scientific Merit Award from the Brazilian Computer Society.



Hélio Lopes is an Associate Professor in the Department of Informatics at PUC-Rio. He has a Doctoral degree in Mathematics (1996), a Master's degree in Informatics (1992) and a Bachelor degree in Computer Engineering (1990). All of these degrees were obtained at PUC-Rio. His research is mainly in the fields of Computer Graphics and Computational Modeling. He likes to develop and to apply advanced mathematical and computational techniques to solve real-world problems that have engineering, scientific and industrial relevance.



Luciana Rosa Redlich has a Master degree in Informatics (2013) and a Bachelor degree in Computer Engineering (2010). All of these degrees were obtained at PUC-Rio. She worked on the area of Multimedia and Hipertext, mainly on projects related to Digital TV.



Jose Antonio F. de Macedo holds his MSc and PhD degree in Computer Science at Pontifical Catholic University of Rio de Janeiro, Brazil. During his PhD course he worked at ISI Lab of ENST-Bretagne, France, and he did a research Database Laboratory at EPFL, Switzerland. From 2006 to 2009, he was a Senior Researcher at Database Laboratory at EPFL, Switzerland. He joined the Computer Science Department of the Federal University of Ceará in 2009, where he is currently an Associate Professor. In 2010, he was granted with a research fellowship from the National Counsel of Technological and Scientific Development (CNPq) of Brazil. He is the author of more than 40 refereed journal and conference papers and his research interests include spatio-temporal data management, cloud computing data management, semantic web and large scale data processing.



Melissa Lemos is an Associated Researcher in the TecGraf Institute at PUC-Rio. She has a Doctoral degree in Informatics (2004), a Master's degree in Informatics (2000) and a Bachelor degree in Computer Engineering (1998). All of these degrees were obtained at PUC-Rio. She has experience in Software Development and Data Bases.



Marcelo Tilio Monteiro de Carvalho is a General Manager in the TecGraf Institute at PUC-Rio. He has a Doctoral degree in Civil Engineering (1995), a Master's degree in Civil Engineering (1990) obtained at PUC-Rio. And its Bachelor degree in Civil Engineering (1990) was obtained in UERJ. He is the principal investigator of several industrial projects in the field of environment and security.



Chiara Renso received the MSc and the PhD in Computer Science from University of Pisa in 1992 and 1998 respectively. She held a permanent research position at ISTI, Institute of National Research Council, Pisa, Italy. Her research interests range from spatio-temporal reasoning, semantic queries on geographical information systems, data mining query languages, semantic and conceptual aspects in data mining, data mining for e-Health, data mining for mobility data, semantic data mining for mobility data. She served as reviewer for several data mining conferences. She is author of more than 100 scientific publications. She is co-editor of a book (*Mobility data: Modelling Management and Understanding*) published in 2013 by Cambridge Press). She has been co-chair of the *ICLP Workshop on Complex Reasoning on Geographical Data (2001)*, of the *Spatial, Temporal, and Spatio-temporal Data Mining (SSTD08)* and the *First, Second and Third International Workshop on Semantic Aspects in Data Mining (SADM)*, held in conjunction with IEEE ICDM conference. She also served as Local Organizer Chair for *IEEE ICDM 2008*. She has been the scientific coordinator of a Bilateral project CNR-CNPQ between the Italian National Research Council and the Brazilian research organization CNPQ in 2012–2014. She is currently the scientific coordinator of the FP-7-PEOPLE project called SEEK (www.seek-project.eu) from 2012 to 2015.