CLEI ELECTRONIC JOURNAL, VOLUME 20, NUMBER 1, PAPER 4, APRIL 2017

Evaluation of Entity Recognition Algorithms in Short Texts

Raquel Fonseca Canales
Universidad de Costa Rica, Posgrado en Computación e Informática
San José, Costa Rica
raquel.fonseca@ecci.ucr.ac.cr

and

Edgar Casasola Murillo
Universidad de Costa Rica, Escuela de Ciencias de la Computación e Informática
San José, Costa Rica, 10501
edgar.casasola@ucr.ac.cr

Abstract

One of the major consequences of the growth of social networks has been the generation of huge volumes of content. The text that is generated in social networks constitutes a new type of content, that is short, informal, lacking grammar in some cases, and noise prone. Given the volume of information that is produced every day, a manual processing of this data is unpractical, causing the need of exploring and applying automatic processing strategies, like Entity Recognition (ER). It becomes necessary to evaluate the performance of traditional ER algorithms in corpus with those characteristics. This paper presents the results of applying AlchemyAPI y Dandelion API algorithms in a corpus provided by The SemEval-2015 Aspect Based Sentiment Analysis Conference. The entities recognized by each algorithm were compared against the ones annotated in the collection in order to calculate their precision and recall. Dandelion API got better results than AlchemyAPI with the given corpus.

Keywords: Information Retrieval, Entity Recognition, ER, Named Entity Recognition, NER, corpus, precision, recall.

1 Introduction

Nowadays social networks like Facebook and Twitter have become a channel for communication and interaction, constituting a powerful source of information for evaluation and prediction of large-scale facts [1]. For example, [2] capture large-scale trends on consumer confidence and political opinions in tweets. Given the volume of information that is produced every day, a manual processing of this data is unpractical, causing the need of exploring and applying automatic processing strategies, like Part-of-speech (PoS) tagging and Entity Recognition.

Traditionally, Named Entities Recognition research has focused on formal text like web pages [3], with some researches on informal text like electronic mails, blogs and clinical notes [4]. The text that is generated in social networks constitutes a new type of content, that is short, informal, lacking grammar in some cases, and noise prone [5]. Due to these characteristics, it becomes necessary to evaluate the performance of traditional ER algorithms in corpus with those characteristics.

This paper focused on the study of algorithms already implemented for Entity Recognition, from which two were chosen for evaluation. The corpus used was from The SemEval-2015 Aspect Based Sentiment Analysis (SE-ABSA15) conference [6]. This corpus has two sets of comments in English, and includes the entities previously identified by the creators of the collection [7]. The entities recognized by each algorithm were compared against the ones annotated in the corpus in order to calculate their precision and recall, to determine which algorithm performs best with the given corpus.

The paper is organized as follows. First, concepts related to the Named Entity Recognition field, like general approaches and relationship with Sentiment Analysis are introduced. Then, section 3 outlines some related work. Sections 4, 5, 6 and 7 describe the corpus, the algorithms selection process, the experiments design and their execution. Section 8 shows an analysis of the obtained results and section 9 focuses on special cases. Finally, future work and conclusions are presented on sections 10 and 11.

2 Named Entity Recognition

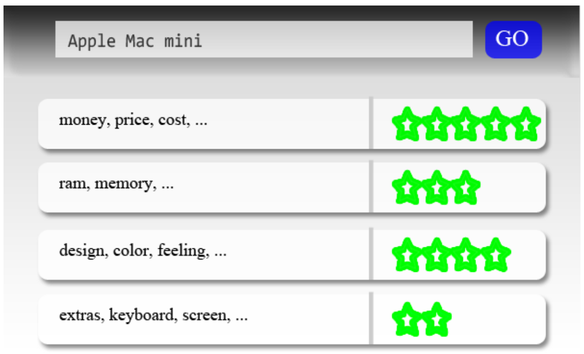
Named entities are information units like names, including people, organizations, places, and numerical expressions like time, dates, money and percentage expressions. Named Entity Recognition is a subtask of Information Retrieval field whose objective is identify references to named entities within a text [8]. For example, in the sentence "Saul is the best restaurant", "Saul" is a named entity.

Named Entity Recognition has a pivotal role in two well-known Natural Language Processing (NLP) tasks: sentiment analysis and topic detection [9]. Sentiment analysis, also known as opinion mining, consists of assigning a sentiment, from a set of possible values, to a given portion of text. Topic detection assigns a class or topic, from a set of possible predefined classes, to a given document [1].

Sentiment analysis studies people's opinions. In general, opinions can be expressed about anything, for example, a product, service, person, event, organization or topic. In this context, the term *entity* is used to denote the target object that has been evaluated. An entity has a set of components (or parts) and also a set of attributes. The term *aspect* is then used to denote both components and attributes. An opinion can be expressed on any component and on any attribute of the entity [9].

An opinion is a positive or negative sentiment, attitude, emotion or appraisal about an entity or an aspect of the entity from the author of the review. An opinionated text can contain positive and negative aspects of the entity, although the general sentiment on the entity may be positive or negative. Aspect-Based Sentiment Analysis (ABSA) extracts aspects that have been evaluated and determines whether the opinions on those aspects are positive, negative or neutral [9].

With Aspect-Based Sentiment Analysis it is possible to generate summaries of all the aspects of an entity and their overall sentiment. For expressing the overall sentiment of an aspect, a rating from 1 to 5 stars can be used. For example, a review with 4 or 5 stars is considered a positive review, a review with 1 or 2 stars is considered a negative review and a review with 3 starts is considered a neutral review [9]. Fig. 1 shows an example of a table listing all evaluated aspects of the "Apple Mac mini" entity and their overall sentiment using a 1 to 5 stars rating [6].



money, price, cost, ...	★★★★★
ram, memory, ...	★★★
design, color, feeling, ...	★★★★
extras, keyboard, screen, ...	★★

Figure 1: Example of a table showing all evaluated aspects of the "Apple Mac mini" entity and their overall sentiment using a 1 to 5 stars rating, taken from [7]

There are three general approaches for performing Named Entity Recognition [10]:

1. Based on rules.
2. Automatic learning.
3. Hybrid methods.

Each approach is briefly described below.

2.1 Based on Rules

Based on rules NER algorithms apply basic techniques, like regular expressions, dictionaries and templates for detecting named entities [11]. In some cases, it is possible to detect a pattern that some entities follow. This pattern can be expressed in a regular expression and then used for recognizing other entities of the same nature. DNI, dates and numbers are examples of entities that follow a pattern [12].

Other technique in this approach is using dictionaries, also known as gazetteer, lexicon and list [8]. Nadeau and Sekine in [8] present three significant categories of dictionaries used in literature: general dictionary, dictionary of entities and dictionary of entity cues. A general dictionary contains common nouns, a dictionary of entities contains specific entities like organizations, countries and states, while the dictionary of entity cues contains typical words, for example, in a person title, location or cardinal point.

Another technique constitutes the use of templates. For example, a template for detecting a person can include a list of attributes like name, last name, DNI. All these attributes will be recognized as entities individually [12].

2.2 Automatic Learning

Automatic machine learning is an alternative to handcrafted rules for automatically induce rule-based systems or sequence labeling algorithms starting from a collection of training examples [10]. With an automatic learning approach the NER system will have the ability to recognize previously unknown entities. This process can be possible through recognition and classification rules that are triggered by features associated with positive and negative examples. There are three learning methods: supervised machine learning (SL), semi-supervised learning (SSL) and unsupervised learning (UL) [8].

The supervise learning method typically consists of a system that studies the features of positive and negative examples of named entities over a large annotated corpus and designs rules that capture instances of a given type. SL techniques include Hidden Markov Models (HMM), Decision Trees, Maximum Entropy Models (ME), Support Vector Machines (SVM), and Conditional Random Fields (CRF) [8]. A system using semi-supervised learning works with a smaller number of examples. It analyzes the context where those entities appear and tries to identify contextual clues. With this information the system will try to recognize other entities that appear in a similar context. Bootstrapping is the main technique for semi-supervised learning, it involves a small degree of supervision, such as a set of seeds, for starting the learning process [8]. The unsupervised learning relies on lexical resources, lexical patterns and on statistics computed on a large unannotated corpus. Clustering is the typical approach in UL, which tries to gather named entities from clustered groups based on the similarity of context [8].

2.3 Hybrid Methods

Hybrid methods use a combination of the first two approaches. For example, a hybrid system could include a module that uses handcrafted rules to propose candidate entities. Then, for retaining the desirable candidates, a decision tree learner could be used to automatically infer a classifier [13].

3 Related Work

Traditionally, Named Entities Recognition research has focused on formal text like web pages [3], with some researches on informal text like electronic mails, blogs and clinical notes [4]. For example, the system developed by Krupka and Hausman in [11], started as an application for processing news stories. The system then moved beyond proper name recognition and news applications to topic categorization or summarization, allowing the analysis of classified ads and email monitoring.

In [13], Janshe and Abney discuss techniques for extracting relevant information from voicemail transcripts. In their research they focused on identifying basic information like the name of the caller/sender or a phone number for returning calls. In [14] the authors explore the use of entity recognition within the Digital Humanities context. They analyze the possibility to derive more value out of existing unstructured metadata from objects documented in the online database of the Smithsonian Cooper-Hewitt National Design Museum.

Research on NER for social networks is a recent direction for the research community. In [15] a multilingual NER system designed for processing Twitter content is described. The system uses SVM and HMM to derive language independent features to model the segmentation and classification tasks. They experiment with three types of features: word, n-gram and context features. A word feature essentially memorizes words from the training set. The n-gram feature identifies the root, prefix, and suffix of a word. A context feature identifies common word patterns to the left or right of a named entity that can be used to identify similar entities of that type.

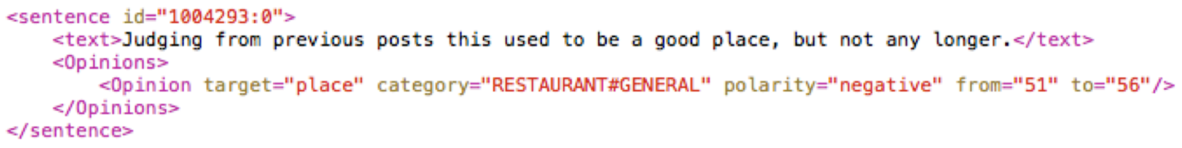
Liu et al. in [5] propose a two stage approach for recognizing named entities in tweets. Their system combines a K-Nearest Neighbors (KNN) classifier with a linear Conditional Random Fields model under a semi-supervised learning framework. The KNN classifier captures global coarse evidence while the CRF model fine-grained information encoded in a single tweet. The KNN incorporates evidence from new labeled tweets and retrains the system. The CRF can output the probability of a label sequence, which can be used as the label confidence.

Batista and Ribeiro present in [1] a strategy based on binary maximum entropy classifiers for automatic sentiment analysis and topic classification over Spanish Twitter data. The adopted approach provides a way of expressing and combining different aspects of the information, and implements feature selection. Binary classifiers were used for discriminating between six possible sentiment classes, and a one-vs-all strategy was used for topic classification, where the most probable topics for each tweet were selected.

4 Corpus

The corpus selection was defined as a fixed parameter of this research: the corpus from the SemEval-2015 conference. This decision was made taking into consideration that the research was motivated because of its use in SA. Experiments described in this paper use English data provided in the context of the Task 12: Aspect Based Sentiment Analysis of the Sentiment Track of the International Workshop on Semantic Evaluation (SemEval-2015). The SemEval-2015 Aspect Based Sentiment Analysis (SE-ABSA15) task focuses on Aspect-Based Sentiment Analysis [7].

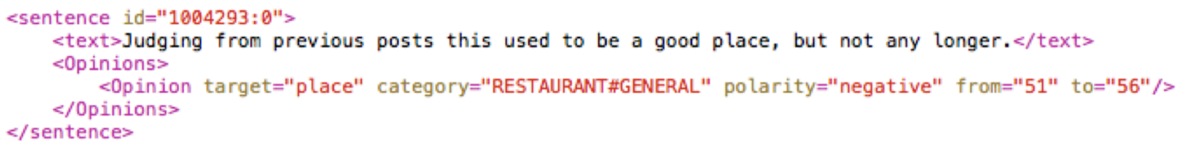
The conference corpus includes two domains: restaurants and laptops, but just the restaurants domain contains information of the entities in the text. The corpus contains two data sets of comments made by people on reviews on the domain. One of these data sets corresponds to training data while the other constitutes the testing data. The files are in XML format, and include annotation data from experts. Fig. 2 shows an excerpt of the restaurants domain. In the XML, each comment and its annotation data are grouped in a *sentence* tag. The *text* tag includes the comment text. Each aspect evaluated in the comment is annotated within the *Opinions* tag using a single *Opinion* tag. The *polarity* attribute indicates the sentiment assigned to the text. The *category* attribute will be covered in detail later.



```
<sentence id="1004293:0">
    <text>Judging from previous posts this used to be a good place, but not any longer.</text>
    <Opinions>
        <Opinion target="place" category="RESTAURANT#GENERAL" polarity="negative" from="51" to="56"/>
    </Opinions>
</sentence>
```

Figure 2: Excerpt of the restaurants domain XML showing a single comment text and its corresponding annotation data

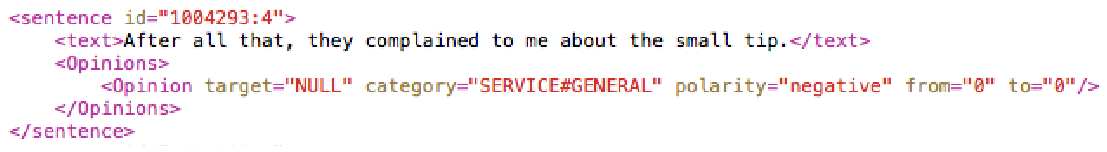
The restaurants data set has opinions expressed about specific entities and their attributes. The entity that is evaluated can be the restaurant as a whole (e.g. restaurant, Saul, Red Eye Grill), its ambience and location, the food and drinks that are served, etc. [6]. The annotation created by the expert includes the Objective Target Expression (OTE). An OTE is an explicit reference (mention) to an entity. Such mentions can be named entities (e.g., "The Four Seasons"), common nouns (e.g., "place", "steak", "bed") or multi-word terms (e.g., "vitello alla marsala", "conference/banquet room") [7]. This reference is identified in a unique form within the text by its start and end position. As shown in Fig. 3, in the XML, the *target* attribute contains the OTE while the *from* and *to* attributes identify its start and end position in the text.



```
<sentence id="1004293:0">
    <text>Judging from previous posts this used to be a good place, but not any longer.</text>
    <Opinions>
        <Opinion target="place" category="RESTAURANT#GENERAL" polarity="negative" from="51" to="56"/>
    </Opinions>
</sentence>
```

Figure 3: Excerpt of the restaurants domain XML showing the OTE and its start and end position within the text

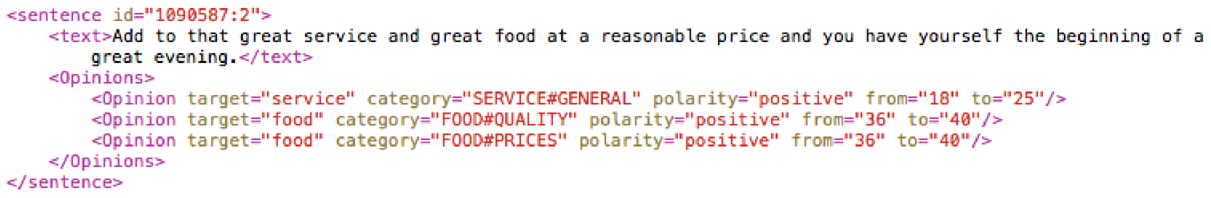
It is important to note that when the entity is implicitly referred (e.g. with pronouns), or it is inferred in a sentence, the OTE is assigned a "NULL" value in the annotation, as is shown in Fig. 4. Because there is no an explicit reference to the entity in the text, the *target* attribute takes a “NULL” value and the *from* and *to* attributes take a zero value.



```
<sentence id="1004293:4">
    <text>After all that, they complained to me about the small tip.</text>
    <Opinions>
        <Opinion target="NULL" category="SERVICE#GENERAL" polarity="negative" from="0" to="0"/>
    </Opinions>
</sentence>
```

Figure 4: Excerpt of the restaurants domain XML showing the *target* attribute in “NULL”

There are other cases where the same entity is present in more than one annotation, this due to different opinions expressed in different aspects. Fig. 5 shows an example of this case, there are two annotations for the *food* entity. It is known that the annotations are referring to the same entity because the *from* and *to* attributes have the same value in both annotations, “36” and “40” respectively. The difference is found in the category that was assigned in the annotations. For the first annotation the “FOOD#QUALITY” category was assigned while “FOOD#PRICES” was the category assigned to the second annotation.



```
<sentence id="1090587:2">
    <text>Add to that great service and great food at a reasonable price and you have yourself the beginning of a
        great evening.</text>
    <Opinions>
        <Opinion target="service" category="SERVICE#GENERAL" polarity="positive" from="18" to="25"/>
        <Opinion target="food" category="FOOD#QUALITY" polarity="positive" from="36" to="40"/>
        <Opinion target="food" category="FOOD#PRICES" polarity="positive" from="36" to="40"/>
    </Opinions>
</sentence>
```

Figure 5: Excerpt of the restaurants domain XML showing two different annotations for the same *food* entity

As it was mentioned before, the corpus contains two datasets for the restaurants domain. Table 1 shows the data contained on each of the data files.

Table 1: Data contained on each of the data files for the restaurants domain

Data	Reviews	Comments	Annotations		
			Total	“NULL”	Multiple
Training	254	1315	1654	375	82
Test	96	685	845	248	55

The corpus annotations also include information about the aspect category. In Fig. 3 this information is contained in the *category* attribute of the *Opinion* tag in the form of a E#A pair, where E corresponds to an entity type and A to an attribute label. The entity type and attribute label are chosen from a predefined inventory. Table 2 shows the list of entity types for the restaurants domain.

From the two domains, just the restaurants data set contains information of the entities that are mentioned in the text. Because of this characteristic the restaurants domain was chosen as the Gold Standard Corpus (GSC). Once considered as GSC, this set is taken as absolute reference about what is a valid entity and what is not. Entities extracted by the algorithms that do not belong to the valid entities set will be considered as an error. Once the corpus was defined the next step was to review Entity Recognition algorithms and choose two for analyzing their results with the corpus.

5 Algorithms Selection

Information about algorithms implemented for ER was collected, reviewing previous work and tools available online. Two algorithms were chosen, prioritizing algorithms that had finished implementations available and in use, that were available online and offered a free access.

Table 3 presents a summary of the systems that were reviewed. Perez and Cardoso [12] developed a system that includes the corpus processing task as well as the corpus analysis. The processing task was developed for working with rectory resolutions in files with different text formats. The analysis phase includes the ER task and a categorization process. This system was not selected for this research because the ER task is not available as an individual component but it is part of a whole system developed for the particular domain.

Table 2: Entity type inventory for the restaurants domain, taken from [7]

Entity Type
FOOD
DRINKS
SERVICE
AMBIENCE
LOCATION
RESTAURANT

It is important to note that given that the system uses an automatic learning approach, a large collection of training annotated data is required.

O'connor et al. [2] analyze Twitter text for measuring their sentiment and compare the results with data from public polls. Given the size of their data (millions of text messages per day), they decided to use a deterministic approach, based in linguistic knowledge, counting words with positive sentiment and those with negative sentiment, for determining the general opinion over a topic. Given that this research focuses on the entity extraction in general, this system could not be selected.

Batista and Ribeiro [1] also analyze Twitter text but in Spanish. They developed a system for automatic sentiment analysis and topic classification, using binary classifiers. Given the strategy used for developing the system, there is no a specific module for the ER, so it could not be used for the purpose of this research.

Krupka and Hausman [11] describe the use of the NetOwl Extractor in the NER task of the MUC-7. For entity extraction the system uses ontologies that include people, organizations, places, address, artifacts, phone numbers, dates and others. The system has been trained with data from various social platforms. NetOwl Extractor is a commercial software product that is licensed, so there is no a free access available online.

Janshe and Abney [13] analyze text manually transcribed from voice mail messages. They use a decision tree for the tagger that identifies the entities related to the caller. For extracting the phone number, they use manually created rules that identify candidate entities and then a decision tree for automatically infer a classifier. Janshe and Abney system points towards approaches that uses a small inventory of characteristics that have been adapted to specific tasks. For this research this system could not be used because the ER will be applied for entities different than the caller information and the phone number found in a voicemail message.

Van Hooland et al. [14] in their work use unstructured metadata from documented objects stored in the online database of the Smithsonian Cooper-Hewitt Museum of National Design. For ER they use tree services: AlchemyAPI [16], DBpedia Spotlight [17] and Zemanta. Zemanta service was found online as Dandelion API [18]. These tree services provide implementations in production, that are used by different organizations, are available online and offer a free access. It was noticed that Dandelion API currently makes use of DBpedia in their entity extraction API so DBpedia Spotlight service is included in Dandelion API.

Google announced in march of the current year about an expansion that they will be doing in their cloud platform with the incorporation of APIs based on machine learning. One of the APIs, Cloud Natural Language [20], is dedicated to the processing of data without formal structure and includes ER. This API was published in beta until July 20th, so it was not possible to include it in this research.

Most of the systems found in recent literature have been developed using the automatic learning or hybrid approaches. In general, these algorithms specialize in the corpus used for its development. The tree general purpose services that were found available online and that offer a free access were AlchemyAPI, DBpedia Spotlight and Dandelion API. However, as Dandelion API uses DBpedia, just AlchemyAPI and Dandelio API were chosen. The source code for these services is not open source so their algorithms could not be inspected and compared.

5.1 AlchemyAPI

AlchemyAPI provides different services for natural language processing through Application Programming Interfaces (API), one of those for entity extraction. In its documentation the service indicates that is capable of identifying people, companies, organizations, cities, geographic characteristics and other entities with type in HTML, text or web based content. For analyzing the information, statistics algorithms and natural language processing technology are used for extracting the semantic included in the text [16]. AlchemyAPI's text analysis functions include: Entity Extraction, Sentiment Analysis, Keyword Extraction, Concept Tagging, Relation Extraction, Taxonomy Classification, Author Extraction, Language Detection, Text Extraction and

Table 3: ER reviewed systems summary

System	Corpus	ER Approach	Characteristics
Perez & Cardoso [12]	Rectory resolutions	Supervised Learning	System developed for the specific domain
O'connor et al. [2]	Twitter messages in English	Based on rules	OpinionFinder is used as lexicon for determining positive and negative sentiment words
Batista & Ribeiro [1]	Twitter messages in Spanish	Supervised Learning	System developed for automatic sentiment analysis and topic classification
NetOwl Extractor [11]	N/A	Hybrid	Commercial software product
Jansche & Abney [13]	Transcribed voicemail messages	Hybrid	Different models are used depending on the information that needs to be extracted
Van Hooland et al. [14]	Objects metadata from the Smithsonian Cooper-Hewitt Museum of National Design	Hybrid	Access to different services
AlchemyAPI [16]	N/A	Hybrid	Online service
DBpedia Spotlight [17]	N/A	Hybrid	Online service
Dandelion API [18]	N/A	Hybrid	Online service
Google Cloud Natural Language API [19]	N/A	Hybrid	Online service

Feed Detection [16]. AlchemyAPI offers free access to its services for research and nonprofit purposes.

5.2 Dandelio API

Dandelion API provides different services for semantic analysis of text. In its documentation it is indicated that the API for entity extraction allows to find places, peoples brands and events, in both documents and social network content, reinforcing its ability to work well in short texts [18]. Dandelion API's text analysis features include: Entity Extraction, Text and Content Classification, Sentiment Analysis, Keywords/Concepts Extraction, Semantic Similarity and Article Extraction [18].

The service provided by Dandelion API performs a high quality identification of entities that are linked in well know data sets in the Linking Open Data (LOD) cloud, like DBpedia or Freebase. Like AlchemyAPI, Dandelio API offers a free access for research and nonprofit purposes.

6 Experiments Design

Retrieval system performance is usually measured by using recall and precision values, where recall is the ability of the system to present all relevant items, while precision is the ability to present only the relevant items [21]. The standard recall R and the standard precision P may be defined as

$$R = \frac{\text{number of items retrieved and relevant}}{\text{total relevant in collection}} \tag{1}$$

$$\text{and} \quad P = \frac{\text{number of items retrieved and relevant}}{\text{total retrieved}}. \tag{2}$$

F-score is other measure commonly used in information retrieval and information extraction evaluations, which calculates the harmonic mean of recall and precision [22], using a value of $B = 1$, the F_1 is defined as

$$F_1 = \frac{2 * \text{precision} * \text{recall}}{\text{precision} * \text{recall}}. \tag{3}$$

In this research the recall and precision metrics were adapted as shown in (4) and (5)

$$R = \frac{\text{number of entities recognized and annotated}}{\text{total entities annotated in collection}} \tag{4}$$

$$\text{and} \quad P = \frac{\text{number of entities recognized and annotated}}{\text{total entities recognized}}, \tag{5}$$

where recall is the percentage of the entities annotated in the corpus that were found by the system while precision is the percentage of entities found by the system that are correct. A named entity is correct only if it is an exact match of the corresponding entity annotated in the corpus XML data file. For example, in the comment "The pastas are incredible, the risottos (particularly the sepia) are fantastic and the braised rabbit is amazing.", one of the annotations in the collection is "braised rabbit" as OTE, with a start position of "87" and end position of "101". One recognized entity by an algorithm is correct if its OTE is an exact match of the "braised rabbit" text, and if its start and end positions matches the ones annotated in the corpus. With this definition, partial matches are taken as error. For example, if the entity "rabbit" is recognized by one of the algorithms, it is not taken as correct when compared to the "braised rabbit" annotated entity.

After the algorithms were selected, each one was used for extracting entities in the chosen corpus. The entities recognized by each algorithm were compared against the ones annotated in the corpus in order to calculate the precision, recall, and F-score, to determine which algorithm performs best with the given corpus.

7 Experiments

For accessing the services, OpenRefine (Freebase Gridworks and Google Refine) [23] was first used. OpenRefine is an Interactive Data Transformation (IDT) tool, that provides access to different NER services through an extension. The advantages of using such type of tools is that they allow to perform operations in large sets of data, for example, reading from different sources, including XML files. Using a ER extension also allows two levels of automatization: just one interaction from the user is required to perform ER operations in multiple records and each record can be analyzed by multiple ER services at the same time [14]. OpenRefine provides a graphic user interface. When the restaurants dataset XML file was processed the data was displayed in a table, showing each comment text in a row. The ER extension created a column in the comments table for showing the results of each ER algorithm used to analyze the data.

After a preliminary test with OpenRefine, a custom tool for accessing the ER services was developed, mainly because the implementation of the ER extraction in OpenRefine abstracts each ER service in a uniform interface, and the results would have to be processed in a special way in order to calculate the required metrics. This tool allowed to reduce the number of operations performed for each service and store the results in a more efficient way for evaluating each algorithm.

7.1 Tool Description

The tool was developed using Java as the programming language. Other technologies that were also incorporated include: Spring, Hibernate, XOM, and Flexjson. The application includes two modules: core and web. The core module contains all the functionality required for running the tool as a standalone application. The web module was incorporated for a future development of a web user interface. Fig. 6 shows the architecture design. The core module reads the corpus data from an XML file and stores its contents in a database. This module also accesses each service for performing ER operations; the results are stored in the database and used during the metrics calculation.

7.1.1 Processing Corpus Data

The first functionality developed in the tool was the processing of the corpus data from the XML file provided by the conference. The SemEvalXMLParser class was created for this purpose using the XOM library. According to the XML structure of the restaurants corpus, a set of classes was created for storing its contents as shown in Fig. 7. The *Review* class represents one review in the data set; it has a unique identifier and can contain multiple comments. Each comment is represented by the *Sentence* class. It has its own identifier and the text. Each comment can have multiple annotations; each annotation is represented by the *Opinion* class. It includes the entity that was evaluated in the *target* attribute, the aspect category

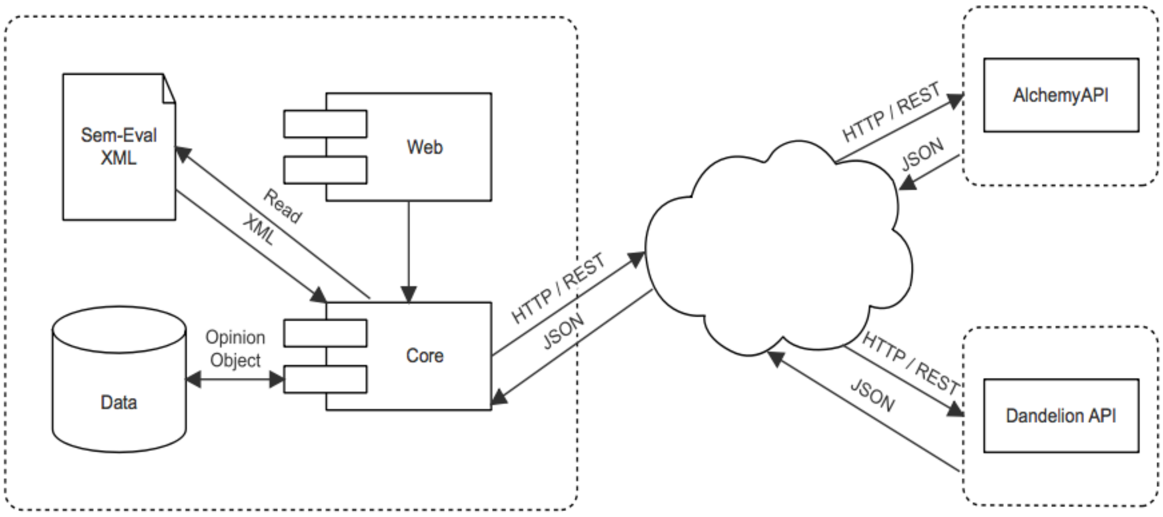


Figure 6: Architecture design showing modules and relationships with data sources and external services

in the *category* field, the sentiment assigned in the *polarity* field, and the *from* and *to* attributes identify the entity's start and end position in the text. Once the information was read from the XML file, it was stored in a relational database, using MySQL as relational database management system.

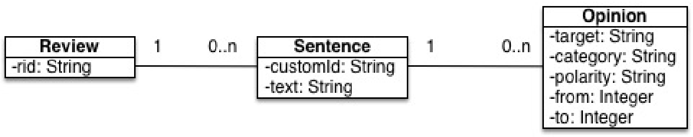


Figure 7: Class diagram with the classes created for corpus reviews

7.1.2 Access to Services

With the corpus data stored in the database, the next step was to access each service for performing ER operations. General classes for representing a ER algorithm, accessing its data and performing ER operations were created so the implementation of the tool allows to add more services in the future. The extensibility would allow future experiments like the use of various ER services for aggregating the results into a single one, with a consensus indicator. Using several ER algorithms simultaneously may enhance the quality of the ER process.

The data flow for performing ER is shown in Fig. 8. Each comment stored in the database is sent to each service for its analysis. For example, assuming the comment "Saul is the best restaurant" was part of the corpus, it would be read from the database and sent to each API for ER. The API will return the list of entities identified in the text, e.g. "Saul" and "restaurant" in the previous example, and those entities will be stored in the database for their later use.

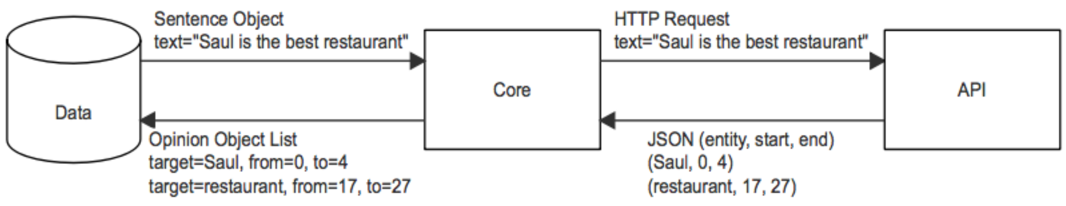

Sentence Object
text="Saul is the best restaurant"
Data
Core
HTTP Request
text="Saul is the best restaurant"
API
Opinion Object List
target=Saul, from=0, to=4
target=restaurant, from=17, to=27
JSON (entity, start, end)
(Saul, 0, 4)
(restaurant, 17, 27)

Figure 8: Data flow for performing ER with each service

During the analysis of each API it was found that both services provide REST web services. The next step was to create classes for querying Alchemy and Dandelion APIs. *AlchemyNerService* and *DandelionNerSer-*

vice classes were created and the Apache HttpComponents library was used for handling the communication between the tool and each service. The parameters required by each service depend on their own API. When accessing AlchemyAPI, the *apikey* and *text* parameters were used as they are required by the API. The *outputMode* parameter was also specified. This parameter is optional and allows to specify the desired response format, in this case, JSON format. The rest of the parameters are optional and were not modified, thus the default values are being used by the API [16]. When querying Dandelion API, the *text*, *$app_id* and *$app_key* parameters were used as they are required by the API. The *lang* parameter was also specified, it is optional and allows to specify the language of the text to be annotated, in this case, English. The rest of the parameters are optional and were not modified so the default values are assumed by the API [18].

For deserializing the JSON returned by each service the *AlchemyJSONParser* and *DandelionJSONParser* classes were created and the Flexjson library was used. These classes specialize in the deserialization of the JSON returned by each algorithm because it depends on each API implementation. The developed tool is currently deserializing all the data returned by the APIs, using classes created for modeling that information, however, for this research, just a subset of the returned fields is required, that is stored in the database using *Opinion* class instances. The *Opinion* class has the *target*, *from* and *to* algorithms, that are required for comparing a corpus annotation with one provided by one of the algorithms. In AlchemyAPI JSON data structure the *entities* field contains the list of entities detected by the API. For each entity, the *text* field includes its text. In Dandelion API JSON data structure the *annotations* field contains the list of entities recognized by the API. For each entity, the *spot* field includes its text.

Once the AlchemyAPI and Dandelio API services were completed, each service was accessed to analyze each comment on the corpus and the entities recognized by each algorithm were stored in the database. With this information the required metrics were calculated. The functionality for calculating the metrics required for this research was included in the *SentenceServiceImpl* class. It accesses the information stored in the database for all the analyzed reviews, and calculates the precision, recall and F-score for each algorithm.

From the 542 entities that were annotated in the corpus, AlchemyAPI identified 19 and Dandelion API 152. In general terms, 168 entities were identified by at least one service. Table 4 shows the precision, recall, and F-score for each service as well as the total and relevant entities identified by each one. Results show that Dandelion API had a better performance than AlchemyAPI in terms of precision and recall, that is also reflected in the F-score metric.

Table 4: Results for each service

Algorithm	Recognized Entities	Relevant Entities	Precision	Recall	F-score
AlchemyAPI	144	19	0.132	0.035	0.055
Dandelion API	**692**	**152**	**0.220**	**0.280**	**0.246**

8 Results Analysis

Analyzing the metrics collected for AlchemyAPI, it is noted that the precision is better than the recall, matching the few entities that were recognized by the service (a total of 144). In the case of Dandelion API, the recall is better than the precision, this could be expected due to the number of entities identified by the service (692 in total). Overall, the recall is below 0.280 for both services, meaning that more than the 70% of the annotated entities by the experts of the SE-ABSA15 conference in the corpus were not identified. Table 5 shows the precision, recall, and F-score, as well as the total and relevant entities identified using both services as a whole. This was done creating one single set with the entities recognized by each service (excluding duplicates). In this case the precision is 0.212, very close to the highest precision gotten when evaluating the algorithms individually (0.220), that is shown in table 4. The recall is 0.310, a higher value than the one gotten when evaluating the algorithms individually (0.280). The F-score metric also improves when using both services combined, from 0.246 to 0.252. The metrics presented in table 4 and table 5 show that just three entities were recognized by both algorithms, given that the number of relevant entities increased from 152 (recognized by Dandelion API) to 168 (16 from the 19 identified by AlchemyAPI).

Table 5: Results combining both services

Algorithm	Recognized Entities	Relevant Entities	Precision	Recall	F-score
AlchemyAPI + Dandelion API	791	168	0.212	0.310	0.252

The entities recognized by each service were analyzed in terms of the category annotated in the corpus. Table 6 shows the number of entities annotated in the corpus by category and the number of entities recognized by each service on each category.

Table 6: Number of entities recognized on each annotated category by each service

Service	AMBIENCE	DRINKS	FOOD	LOCATION	RESTAURANT	SERVICE
SemEval	61	20	272	8	78	103
AlchemyAPI	0	0	6	0	9	4
Dandelion API	9	1	106	0	10	26
Combined	**9**	**1**	**111**	**0**	**17**	**30**

Table 7 shows the recall for each category. The entities annotated as Location category were the most difficult to recognize. None of the services identified any of the entities in this category. The highest recall is obtained in the Food category when using the algorithms as a whole, where it scores 0.408. AlchemyAPI obtained a very low recall in all the categories.

Table 7: Recall scored on each annotated category by each service

Service	AMBIENCE	DRINKS	FOOD	LOCATION	RESTAURANT	SERVICE
AlchemyAPI	0	0	0.022	0	0.115	0.039
Dandelion API	0.148	0.050	0.390	0	0.128	0.252
Combined	**0.148**	**0.050**	**0.408**	**0**	**0.218**	**0.291**

In terms of the results returned by the services, it was noted that each entity comes with a value between 0 and 1, being 1 the most relevant. According to AlchemyAPI documentation [16], this value is a relevance grade while for Dandelion API [18] it is a confidence value of the recognized entity. For AlchemyAPI, there is no parameter for specifying a threshold for this value, but Dandelion API does allow it. Although it is possible to specify a threshold for Dandelion API, the default value 0.6 was used. With the confidence values it is possible to calculate for each service the maximum value, minimum, average and standard deviation. Table 8 presents these results.

Table 8: Metrics calculated for each service using the confidence value

Algorithm	Max value	Min value	Average	Standard deviation
AlchemyAPI	**0.992**	0.010	0.229	0.163
Dandelion API	0.981	**0.607**	**0.738**	**0.072**

In table 8 it is noticed that the confidence values for AlchemyAPI recognized entities are low, with an average of 0.229 and a standard deviation of 0.163. Dandelion API results are different, the average confidence value is 0.738, greater than AlchemyAPI, with a standard deviation of 0.072.

9 Discussion

The obtained results were low and it was not expected. For AlchemyAPI, in its source it is indicated that the entities extraction is based on sophisticated algorithms and natural language processing technology unique in the industry, that combines Multilanguage support, linked data and disambiguation sensitive to context [16]. Regarding DandelionAPI, its source indicates that it has a good performance even on short texts [18].

Van Hooland et al. [14] obtained precisions higher than 0.6 and recalls higher than 0.3 for AlchemyAPI and precisions closer to 0.8 and recalls closer to 0.45 for Dandelion API. It is noticed then a strong dependency on the corpus. This is mentioned by Nadeau and Sekine [8], that the entity recognition problem has a strong dependency on the application domain and that algorithms with a good performance on a domain or language may not perform the same in others.

Analyzing the annotated entities in the corpus it was noted that some of them are composed by more than one term, for example “braised rabbit”. In some cases, the algorithms identified each term individually, for example “braised” and “rabbit”. Due to the scope of this research these cases were not counted nor analyzed in a case by case basis. Work on Named Entity Recognition has usually overlooked nested entities and instead has chosen to focus on the outermost entities. This is a problem that Finkel and Manning discuss in [10], given that most corpus designers have chosen to annotate only the topmost entities.

It is important to remember that for calculating the metrics in this research, an entity recognized by one of the algorithms is taken as correct only if it is an exact match of an entity annotated in the corpus. Other experiments could be performed where a partial match could be used for evaluating a recognized entity as correct, to see if the quality of the results improves.

10 Future Work

When working in the context of the SE-ABSA15 conference, the recognized entities that correspond to individual terms of an annotated entity, were taken as errors. Due to the scope of this research, the entities that were recognized by the algorithms but do not correspond to an annotated entity were not analyzed. It could be possible to perform a manual inspection of those recognized entities to determine their relevance in other context. Another experiment would be to expand the corpus annotations so they include nested entities for the algorithms evaluation.

In Dandelion API documentation there is a parameter that defines the degree on which the Entity Extraction API should rely more on the context or favor more common topics to discover entities. The accepted values for this parameter range between 0 and 0.5, with 0.3 as the default value. The documentation specifies that if a higher value is used, it favors more common topics, that may lead to better results when processing fragmented inputs where the context is not always reliable [18]. For this research, this value was not modified and the default value was used. As future work, experiments could be performed to vary the value of this parameter to see how it affects the precision and recall of the service in the context of comments.

The input datasets provided by the SE-ABSA15 conference also contain information that groups some comments. In this work, that information was not used because the goal was to analyze each comment individually. In a future experiment, the evaluation of the algorithms could be made using those groups as input, given that isolated comments can potentially be out of context.

The extensibility feature of the developed tool allows future experiments, like using various ER services for aggregating the results into a single one, with a consensus indicator. Using several ER algorithms simultaneously may enhance the quality of the ER process.

11 Conclusions

In this paper two implemented algorithms for entity recognition were identified: AlchemyAPI and Dandelion API. For applying each algorithm for entity recognition in the corpus provided by The SemEval-2015 Aspect Based Sentiment Analysis conference, a tool was developed. This tool allows the corpus processing, the access to each service and the metrics calculation.

When evaluating the precision and recall of both services in the given corpus, Dandelion API had a better performance than AlchemyAPI, in the precision as well as in the recall. Nevertheless, due to the low values shown in the calculated metrics, further research is necessary in this area.

Acknowledgment

The authors would like to thank professor Rodrigo Bartels, researcher from the Centro de Investigaciones en Tecnologías de la Información y Comunicación (CITIC), for his expert feedback and contributions on this research. This work has been supported in part by the Universidad de Costa Rica (UCR) and the Ministerio de Ciencia, Tecnología y Telecomunicaciones (MICITT) de la República de Costa Rica.

References

[1] F. Batista and R. Ribeiro, "Sentiment analysis and topic classification based on binary maximum entropy classifiers," *Procesamiento de Lenguaje Natural*, no. 50, p. 77, March 2013.

[2] B. O'Connor, R. Balasubramanyan, B. R. Routledge, and N. A. Smith, "From tweets to polls: Linking text sentiment to public opinion time series," *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[3] D. Downey, M. Broadhead, and O. Etzioni, "Locating complex named entities in web text," *IJCAI'07 Proceedings of the 20th international joint conference on Artifical intelligence*, pp. 2733–2739, 2007.

[4] E. Minkov, R. C. Wang, and W. W. Cohen, "Extracting personal names from email: applying named entity recognition to informal text," *HLT '05 Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 443–450, 2005. [Online]. Available: https://doi.org/10.3115/1220575.1220631

[5] X. Liu, S. Zhang, F. Wei, and M. Zhou, "Recognizing named entities in tweets," *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 359–367, 2011.

[6] SemEval-2015. (2016, January). [Online]. Available: http://alt.qcri.org/semeval2015/

[7] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "Semeval-2015 task 12: Aspect based sentiment analysis," *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015. [Online]. Available: https://doi.org/10.18653/v1/S15-2082

[8] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007. [Online]. Available: https://doi.org/10.1075/li.30.1.03nad

[9] C. C. Aggarwal and Z. ChengXiang, *Mining Text Data.* Springer, 2012.

[10] J. R. Finkel and C. D. Manning, "Nested named entity recognition," *EMNLP*, pp. 141–150, 2009. [Online]. Available: https://doi.org/10.3115/1699510.1699529

[11] G. R. Krupka and K. Hausman, "Isoquest, inc.: Description of the netowltm extractor system as used for muc-7," *MUC-7*, 1998.

[12] M. A. Pérez and A. C. Cardoso, "Técnicas de extracción de entidades con nombre," *XIV Argentine Symposium on Artificial Intelligence (ASAI 2013)*, pp. 109–120, September 2013.

[13] M. Jansche and S. P. Abney, "Information extraction from voicemail transcripts," *EMNLP*, pp. 320–327, 2002. [Online]. Available: https://doi.org/10.3115/1118693.1118734

[14] S. van Hooland, M. D. Wilde, R. Verborgh, T. Steiner, and R. V. de Walle, "Exploring entity recognition and disambiguation for cultural heritage collections," *Literary and Linguistics Computing*, 2013.

[15] D. Etter, F. Ferraro, R. Cotterell, O. Buzek, and B. V. Durme, "Nerit: Named entity recognition for informal text," JHU HLTCOE, Baltimore, MD, USA, Tech. Rep., July 2013.

[16] AlchemyAPI. (2016, January). [Online]. Available: http://www.alchemyapi.com/

[17] DBpedia. (2016, January) Dbpedia spotlight. [Online]. Available: https://github.com/dbpedia-spotlight/dbpedia-spotlight

[18] DandelionAPI. (2016, January). [Online]. Available: https://dandelion.eu

[19] Google. (2016, July) Cloud natural language api. [Online]. Available: https://cloud.google.com/natural-language/

[20] ——. (2016, July) Google cloud platform blog. [Online]. Available: https://cloudplatform.googleblog.com/2016/07/the-latest-for-Cloud-customers-machine-learning-and-west-coast-expansion.html

[21] G. Salton, *Introduction to Modern Information Retrieval*, ser. Mcgraw Hill Computer Science Series. Mcgraw-Hill College, 1983.

[22] C. van Rijsbergen, *Information Retrieval.* MA, USA: Butterworth-Heinemann Newton, 1979.

[23] OpenRefine. (2016, January). [Online]. Available: http://openrefine.org/

[24] Y. Wang, "Annotating and recognising named entities in clinical notes," *ACLstudent '09 Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, pp. 18–26, 2009. [Online]. Available: https://doi.org/10.3115/1667884.1667888

[25] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on- line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997. [Online]. Available: https://doi.org/10.1006/jcss.1997.1504

[26] E. F. T. K. Sang and D. M. Fien, "Introduction to the conll-2003 shared task: Language-independent named entity recognition," *CONLL '03 Proceedings of the 7th Conference on Natural Language Learning*, vol. 4, pp. 142–147, 2003.