EPiC Series in Language and Linguistics

Volume 1, 2016, Pages 97–110

CILC2016. 8th International
Conference on Corpus Linguistics

An interaction approach between services for extracting relevant data from Tweets corpora

Mehdy Dref[1] and Anna Pappa[2]

[1] LIASD, Université Paris 8, Saint-Denis, France
md@ai.univ-paris8.fr
[2] LIASD, Université Paris 8, Saint-Denis, France
ap@ai.univ-paris8.fr

Abstract

We present a system based on the need of special infrastructure adequate to software agents to operate, to compose and make sense from the contents of the Web resources through the development of a multi-agent system oriented services interactions. Our method follows the different construction ontology techniques and updates them by extracting new terms and integrate them to the ontology.It is based on the detection phrases via the ontological database DBPedia. The system treats each syntagme extracted from the corpus of messages and verifies whether it is possible to associate them directly to a DBPedia knowledge. In case of failure, these service agents interact with each other in order to find the best possible answer to the problem, by operating directly in the phrase, trying to semantically modify it, until the association with ontological knowledge becomes possible. The advantage of our approach is its modularity : it is both possible to add / modify / delete a service or define a new one, and then influence the outcome product. We could compare the results extracted from a heterogeneous body of messages from the Twitter social network with Tagme method, based mainly on storage and annotation of encyclopaedic corpus.

1 Introduction

Since the emergence of social networks and real-time public information on the Internet, the continual changes of the Web in the Era of the big data makes the data collection and processing harder by human operators. It's the information monitoring, which characterizes all the implemented strategies that keep us informed about a special subject, in the least time possible and by using automated (signal) reporting processes, which is the most impacted.

Yet this domain is increasingly highlighted by the industry and politics to monitor their ecosystem and ensure the good image/reputation of their brand image or to monitor and adapt their communication strategy.

We consider that the analyst's work requires a special infrastructure adequate to software agents to operate, to compose and make sense from the contents of the Web resources. We demonstrate this need through the development of a multi-agent system oriented interactions between services.

A.Moreno Ortiz and C.Pérez-Hernández (eds.), CILC2016 (EPiC Series in Language and Linguistics, vol. 1), pp. 97–110

Therefore, our contribution follows the different construction techniques and updates ontology proposed in the literature and in particular the called adaptive multi-agent systems. These techniques aim to extract new terms and to integrate them into the ontology [5].

The technique we propose is based on the detection phrases via the ontological database used as DBPedia [1]. To achieve this, we have developed a prototype that incorporates part of the work done by the SMAC team LIFL by developing a multi-scale model for the simulation oriented interactions [3] by moving towards the use of Agents acting as an interface between the system and an online service such as databases of Acronyms, Synonyms, Homonymes. At a global level, the system treats each syntactical group (nominal, verbal) extracted from the corpus of messages and verifies whether it is possible to associate them directly to a DBPedia knowledge. In case of failure, these service agents interact in order to find the best possible answer to the problem, by operating directly in the phrase, trying to semantically modify it, until the association with ontological knowledge becomes possible [6], [7].

If the process doesn't succeed, the phrase is not extracted from the message and is considered irrelevant. The advantage of our approach is its modularity ; the possibility of interactions between services, it is both possible to add / modify / delete a service or define a new one, and then influence the outcome product.

Following the implementation of our prototype, we could compare the results extracted from a heterogeneous body of messages from the Twitter [4] social network with Tagme method [2], which approach is based mainly on storage and annotation of encyclopaedic corpus.

That may become disabling whether considering the changing aspect of Big Data and the needs of industry in order to benefit the real-time information from constantly updating databases as DBPedia.

2 TAGME the On-the-fly Annotation of Short Text Fragments tools

As mentioned in [2] "In the domain of the fast annotation of short messages in social networks, TAGME is a powerful tool that is able to identify on-the-fly meaningful short-phrases (called "spots") in an unstructured text and link them to a pertinent Wikipedia page in a fast and effective way. This annotation process has implications which go far beyond the enrichment of the text with explanatory links because it concerns with the contextualization and, in some way, the understanding of the text."

TagMe uses a snapshot of Wikipedia, from the November 6, 2009. The data are indexed locally. Anchors were drawn from Wikipedia pages, so they added the titles of redirect pages and some variants of the page titles.

"The idea is to manipulate the indexed data, by using for example statistical methods to find the most relevant keyword to enrich the initial short text."

As indicated, TagMe requires at first to recover data from an indexed knowledge database, which is problematic for us in terms of storage in a big data context and in a real-time context.

This is why we implemented an approach using the full capacity of big data, by using services agents communicating, in order to find the best result, taking into account the context and databases that can evolve in real time, and also to provide new exploitable information at any time, or news in real time, like it's the case with Twitter.

In this work, we will explain our ideas by using some examples, and we will compare our approach with Tagme only about the relevance of data. We are not concerned in the scope of the performance in this paper, although we don't forget this important aspect as you will see

in the next parts of this paper.

The main interest of our approach is to provide a system capable of processing lexical ambiguities within a corpus of tweets in order to extract from a data stream relevant and useful messages to build a corpus. Therefore, the method is connected to different APIs developed to retrieve data from a stream on Twitter and allows to analyze, in real-time or not, a source message.

The selection of relevant messages is an important part of our system because it permits to filter a huge amount of data and select a subset of messages to analyze in the next step.

3 Automated Building of a Corpus of Tweets

To introduce our work, it's necessary to define and explain what is Twitter and why this social network is really useful for our work. As mentioned in [4] :

> "Twitter, a microblogging service less than three years old, commands more than 41 million users as of July 2009 and is growing fast. Twitter users tweet about any topic within the 140-character limit and follow others to receive their tweets."

- Number of monthly active users (MAU): 304 million;
- Number of monthly active users who post tweets: 117 million;
- Every minute 350,000 tweets are posted on Twitter, or 183.96 billion tweets per year.

So you understand now why Twitter is a good research study subject to retrieve a huge amount of data. Our system of fast annotation of short messages is also usable for the automated construction of a thematic corpus and we will see how we do that. But before that, we will explain how we retrieve data from the studied social network.

As mentioned below, Twitter developers propose two mains APIs. For our work, we built a third API from scratch by using the Python programming language to retrieve more data without the limitation from the Twitter company.

So, in the case of Twitter, we have for now three types of APIs that we can compare :

- JSON API (limited but allows to retrieve published data);
- Streaming API (interesting to retrieve text in real time without limitation);
- From Scratch API (automatic scrapping that simulates a browser. Good compromise between the two approaches but it needs to be modified when the twitter structure evolves. There is no time or query limitation with our API).

In this step, we can define three ways to retrieve messages. We combined the three APIs to build an Agent in order to retrieve a data stream.

The result produced by this Agent may also be used in others professional contexts, such as the real-time monitoring.

After building this Agent, it's necessary to find a good way to select relevant messages. For this, it's important to define what is "relevant". To illustrate this idea, we can make a real-time search from a hashtag (a keyword related to an event, for example) on Twitter and index in real-time all messages about this event.

If we study for example, the extraction of messages mentioning the President of the United States of America, Barack Obama, the primary idea would be to list all the keywords that can refer to the U.S. President. After that, we just have to detect in the stream or in the feed of messages the presence or absence of these keywords.

Here is a part of this list :

- Barack Obama
- Barack Hussein Obama
- Obama
- President Obama
- The president of the United States of America
- The president of the USA
- Brack Obma (misspelling case)
- @BarackObama (mention on Twitter)
- ...

The main difficulty of this simple approach is to list a complex set of elements relative to the President of the United States of America. So we implemented a system that has the ability to work without this complex list. We simply indicate that we want to use the Barack Obama resource on DBPedia which corresponds to the Barack Obama page on Wikipedia. We can define a resource in DBPedia with a link like in the following table. Our system will be able to associate each type of name relative to Barack Obama with the knowledge database by using several services that we have to define previously.

Keywords	Resource
Barack Obama	http://dbpedia.org/page/Barack_Obama

Table 1: DBPedia Ontological Resource

For example, if I'm interested in tweets of "Barack Obama", I will need agents for:

- messages with one or more spelling mistakes ;
- the message with relatives phrases: President Obama, Obama, President of the United States ...

For this, after extraction of phrases that we want to analyze, a first comparison with data from Wikipedia is performed and if it fails, the problem is transmitted to other services that will try to solve it.

In the case of spelling errors, the system will use a service able to offer spelling corrections like it is the case when performing a Google search. After this step, the corrected phrase will be resubmitted to Wikipedia. Another correction methods are also used in particular, thanks to the modularity of our system.

If that fails, we use a service based on a Google search result that allows to obtain results of an expression such as "President Obama" to retrieve a set of Bi-Grams on the search results that we sort by frequency. We can match via this method "President Obama" with "Barack Obama" that allows the final tweet validate and incorporate our corpus.

In addition, since knowledge databases of several Internet services are regularly updated, if a new name for the President of the United States of America appears, it will be associated with our main resource.

The effectiveness of the system will be determined by the number of services used, but also by the size of the interactions list, and by the order of priority assigned to each interaction.

Our corpus generation system uses other parameters such as the dates of beginning and end, relationships between a speaker and someone who interests us (i.e. people following each other), etc. Since this system is based on the Twitter's API, it is possible to use a certain number of parameters to target messages that can potentially interest us.

Twitter offers the following list of parameters :

- delimited
- stall_warnings
- filter_level
- language
- follow
- track
- locations
- count
- with
- replies
- stringify_friend_id

The progressive dimension of the system through flexible agents permits to adapt our tool for a lot of professional tasks.

4 Our initial approach

After retrieving all the "good" messages, it is necessary to analyze each message to extract data from DBPedia database (or Wikipedia).

Initially our method is divided into three stages. It takes as input a short message and gives in the output a text enriched by links to Wikipedia or DBPedia.

- Split the content (using POS tagging with a from scratch algorithm to split and order the content) ;
- Use knowledge databases to associate data. ;
- Extract data.

To split the content of a message, we use at first a service that produce morpho-syntax graph as known as part-of-speech tagging (POS tagging or POST). It consists in the identification of words as nouns, verbs, adjectives, adverbs, etc., in a message. Also, the part-of-speech tagging works with relationship with adjacent and related words in a phrase, sentence, or paragraph. There is a lot of POS tagging services on the Internet as http://mshang.ca/syntree/ or http://ironcreek.net/phpsyntaxtree/.

In a multilingual context, it will be possible to define as much POS tagging agents as languages. For that, it will be necessary to build for each language a specific precondition as "is_french(word) or is_english(word). The precondition in the interactions list will guide the agent's choices toward the best method.

5 Description of a Simple Agent-Based Service

The smallest element of the system is called a reactive service agent. It is a simple agent that takes an input data as an argument and that connects to a service to retrieve information that will be returned. A simple agent takes as input between 1 and n n-grams and returns the output from 1 to n n-grams. For example, take the case of an acronym agent who takes an acronym parameter and returns a list from 0 to n possibilities.

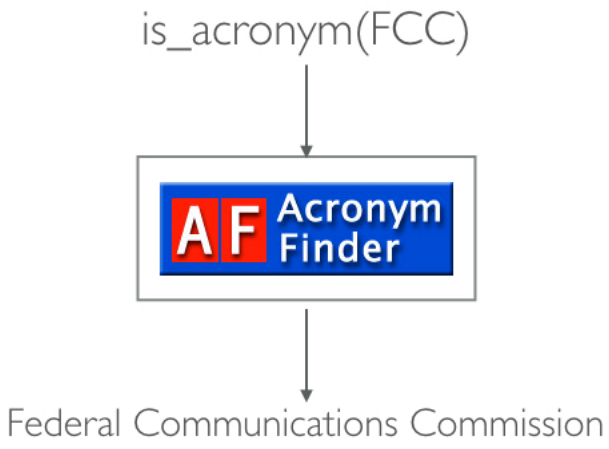

Figure 1: A simple AcronymFinderAgent

6 Description of a Complex Agent-Based Service

In addition to the simple agent, a complex agent has the ability tomake decisions. For this, it has a list of interactions that allows it to choose whom to contact first. Just like the simple agent, it takes as input between 1 and n n-grams and returns the output from 1 to n n-grams. It can internally have an algorithm for processing the data received by the online service to filter the results, for example. This agent may be reactive without limitation, cognitive or hybrid.

Regarding the list of interactions, it is similar to the interaction matrix of the IODA methodology of the SMAC team of the Crystal laboratory of University of Lille 1.

We can define an interaction by a set composed of a target agent and a precondition (is_an_acronym() for example). Consequently, if FOO is an acronym, then we will call the single agent that we defined in the previous section. If necessary, a precondition may itself call a simple service or locally use an algorithm. The modularity of the system allows to compose a multi-agent system, taking into account the constraints of the system developer.

```
#Python function for acronym checking
    def is_an_acronym(word_i_want_to_check):
        upper_case_letters = "QWERTYUIOPASDFGHJKLZXCVBNM0123456789."
        abbrevation = ""
        for letter in word_i_want_to_check:
            if letter in letters:
                abbrevation += letter
            if word_i_want_to_check == abbreviation :
                return True
     return False
```

Regarding the interactions list, there are two common ways to build it : by building a complex agent with interactions to communicate with simple agents, or by building an agent that communicate with others complex agents. The main interest of the second approach is the modularity. Indeed, we can delegate several parts of the system and model subsystems.

Priority	Precondition	Target	Website
1	is_an_acronym	AcronymFinderAgent	http://www.acronymfinder.com/
2	is_an_acronym	AcronymSearchAgent	http://www.acronymsearch.com/
3	is_splitable	SplitStringAgent	None

Table 2: Interactions list for an Agent that communicates with simple agents

Priority	Precondition	Target
1	is_an_acronym	AcronymAgent
2	is_splitable	SplitStringAgent

Table 3: Interactions list for an Agent that communicates with complex agents

Priority	Precondition	Target	Website
1	is_an_acronym	AcronymFinderAgent	http://www.acronymfinder.com/
2	is_an_acronym	AcronymSearchAgent	http://www.acronymsearch.com/

Table 4: Interactions list for a complex AcronymAgent

The main interest of the last method is to wrap all acronym agents mentioned previously, in a specific agent.

To explain how the system works, for each element, the agent will check if the n-gram is an acronym. If not, it will try to split the string to see if any of the items is an acronym.

If an acronym is found, the system will attempt to retrieve the equivalence. And if nothing is found, then the acronym agent will return None and the system will try to split again the string if it is possible. The interactions list will be called again.

We won't discuss more here about complex preconditions that need to introduce the concept of multi-scale multi-agent system oriented interactions between services but we refer you to [7] for an introduction to this important field about our future works.

7 A multi-agent-based model for service-oriented interaction

We consider that any Internet data present can become a service within our system. It can be a simple web service or a more complex service whose results are generated through the result produced, for example, by a search engine, according to algorithms that may be favorable to us. And this is the case for our agent that allows the modification by the substitution using frequency algorithms on Googles search results.

This agent takes into account the fact that Google results are often linked to current events. Therefore, relevant results dealing with current themes appear in the top of the search engine page, and may fill the entire first page.

It is from this observation that we have put in place an agent that can find "synonyms" of the searched term.

To be more specific, lets take the example of Barack Obama, who is the main object of our study in the context of this paper. As you know, there are lots of ways to name the President of the United States. The idea is to understand that a search engine like Google will be able to provide reliable correspondence information (contextually), whatever the term used.

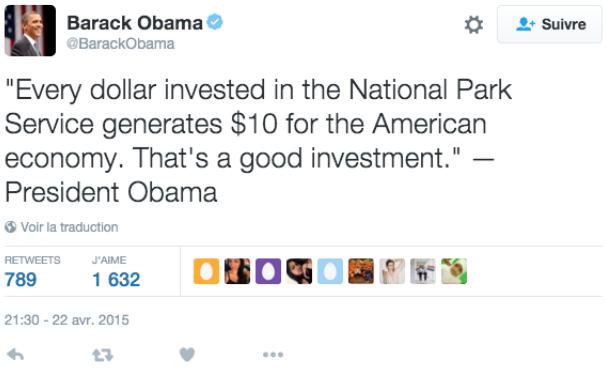

Figure 2: A First Tweet From the US President

If I search for "President Obama" in Google, I notice that on the page is mentioned, Barack Obama. My idea is to measure statistically the presence of the bi-gram Barack Obama, to determine if President Obama means Barack Obama or if it means something else, as you can see in the following figures.

Figure 3: Google research for "President Obama"

Figure 4: Google suggestion (in Wikipedia) for "President Obama"

This allows me to substitute the bi-gram "Barack Obama" to the bi-gram "President Obama". This resolves an ambiguity regarding DBPedia and validates the message.

In the same idea, it is possible to use Google as a spellchecker agent. As you know, Google is able to propose corrections via its suggestion module. Therefore, it is easy to retrieve a correction of a term and to associate it, collecting the proposed data.

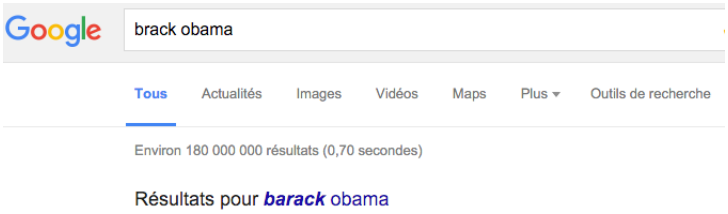

Figure 5: Google as a Service for Fixing Spelling Mistakes

As mentioned in the example, it is possible to find Barack Obama in the page search for "Brack Obma" and for every other research with a spelling mistake. We can then propose a spelling checker agent which is also based on Google.

To conclude this section, one should know that a web service can be understood broadly and that a web page is able to provide a multitude of exploitable services, in order to have relevant data, updated in real time.

8 Example and comparison with TagMe

We will take as an example a message posted by the President of the United States, Barack Obama, about the net neutrality and the FCC.

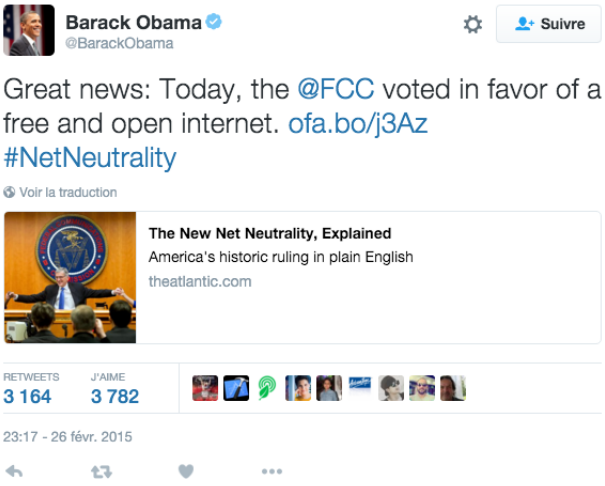

Figure 6: A second tweet from the US President

By using the segmentation algorithm, we can get the following :

```
(ROOT
(S
(NP (DT the) (NN @FCC))
(VP (VBD voted)
(PP (IN in)
(NP
(NP (NN favor))
(PP (IN of)
(NP (DT a)
(ADJP (JJ free)
(CC and)
(JJ open))
(NN internet))))))))
```

Figure 7: A morpho-syntax graph from the tweet

Therefore, we can define a list of elements to analyze by priority order.

In some cases, we have to remove some elements. It is the case, for example, when we have a stop word (a stop word can be found with a dedicated service). In this case, we remove "of" and "a" because there is an n-gram next to them. Also, we remove the "in" between "voted"

and "favor", because "voted" is a verb. We can also remove "the", because it's a stop word and we have a mention next to it. That means that we have to treat the mention separately, because it is destined to someone in particular.

FCC	Voted	Favor	free and open internet

Table 5: 1st step of the analyze

Free	open internet

Table 6: second step of the analyze

Our Approach	Tag Me
News	Hypertext Transfer Protocol
Federal Communications Commission	Free software
Voting	the Internet
Open internet	The News Today
The Atlantic	Debian
Net neutrality	Orthopedic Foundation for Animals
	Federal Communications Commission
	Network neutrality

Table 7: Comparison of results

The link will be analyzed in the same way. We won't describe the algorithm for obvious reasons, but you can find in the next lines how we treated this kind of data. If the entity is a link, then our system will use an agent that will analyze the page. The agent will extract a set of data that will seem relevant by the system designer, which we define below:

- content of the title tag
- content of the h1 tag
- content of the Meta description tag
- content of the first h2 tag
- the link will be ignored

In general, unlike TagMe, our approach permits to gain in quality. This quality is gained by three ways : either by proposing an identical extraction, either by proposing a more specific extraction, or by proposing a reduced extraction (by deleting non-relevant elements, for example).

In terms of performance, we are not able for now to evaluate our approach compared to TagMe, and as indicated in the beginning of this paper, this is not our goal.

9 Conclusion and outlook

To conclude, after the implementation of our prototype, we were able to compare the results extracted from a heterogeneous corpus of messages from the Twitter social network with the TagMe method. Remember that the TagMe approach is based primarily on storage and annotation with an encyclopaedic corpus, which can be outdated if we consider the changing aspect of Big Data and the needs of the industry to get information and knowledge in real time.

Additionally, about industrialists needs, it will be quite possible to consider a system that will be linked with the companys products catalog, with customers, or suppliers, instead of a knowledge database as DBPedia. This system will be able to propose an extraction of relevant messages that will be analyzed and exploited as part of a business intelligence. Therefore, the company will be able to adapt its supply and demands to the target market.

Finally, the strength of our approach is based on its modularity, undoubtedly. By proposing an adjustable approach based on a multi-agent system, the system will be able to be adapted to several problematic, going beyond social networks. Moreover, comparative approaches will be led to determine for every agent the perfect service and the perfect algorithm, depending on needs, and that will be the object of our future works.

10 Future Work

First, one of our future works is to integrate a "feeling" agent to evaluate the sentiment expressed in a message and therefore extract, for example, only negative messages that allow to model a precise aspect of a brand's eco-system, for example. So it will be possible to retrieve in a stream only messages with positive opinion.

Next, as mentioned, we also work in a Multi-scale paradigm. In the future, we will implement some ideas to improve our system.

And finally, as indicated at the beginning of this paper, we want to work with the ontological knowledge database to produce a specific and thematic domain ontology from social network data included without limitation, opinion, relationship between users, influence. For that, we will define an ontological template to represent the social media communication on the Internet. Our system will be able to generate knowledge from social networks extended by DBPedia knowledge.

References

[1] Sren Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *In 6th Intl Semantic Web Conference, Busan, Korea*, pages 11–15. Springer, 2007.

[2] Paolo Ferragina and Ugo Scaiella. Tagme: On-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 1625–1628, New York, NY, USA, 2010. ACM.

[3] Yoann Kubera, Philippe Mathieu, and Sébastien Picault. IODA: An interaction-oriented approach for Multi-Agent Based Simulations. *Journal of Autonomous Agents and Multi-Agent Systems*, 23(3):303–343, 2011.

[4] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

[5] Zakaria Maamar, Soraya Kouadri Mostefaoui, and Hamdi Yahyaoui. Toward an agent-based and context-oriented approach for web services composition. *IEEE Transactions on Knowledge and Data Engineering*, 17(5):686–697, 2005.

[6] Kvin Ottens, Nathalie Hernandez, Marie-Pierre Gleizes, and Nathalie Aussenac-Gilles. A Multi-Agent System for Dynamic Ontologies. *Journal of Logic and Computation, Special Issue on Ontology Dynamics*, 19:831–858, 2009.

[7] Sébastien Picault, Philippe Mathieu, and Yoann Kubera. PADAWAN, un modèle multi-échelles pour la simulation orientée interactions. In M. Occello et L. Rejeb, editor, *JFSMA'2010 – Mahdia (Tunisie) – 18-20 octobre 2010*, pages 193–202, France, 2010. Cépaduès.