Can we use Linked Data Semantic Annotators for the Extraction of Domain-Relevant Expressions?

Michel Gagnon
Ecole Poytechnique de Montréal
michel.gagnon@polymtl.ca

Amal Zouaq
Royal Military College of Canada
amal.zouaq@rmc.ca

Ludovic Jean-Louis
Ecole Poytechnique de Montréal
ludovic.jean-louis@polymtl.ca

ABSTRACT

Semantic annotation is the process of identifying expressions in texts and linking them to some semantic structure. In particular, Linked data-based Semantic Annotators are now becoming the new Holy Grail for meaning extraction from unstructured documents. This paper presents an evaluation of the main linked data-based annotators available with a focus on domain topics and named entities. In particular, we compare the ability of each tool to annotate relevant domain expressions in text. The paper also proposes a combination of annotators through voting methods and machine learning. Our results show that some linked-data annotators, especially Alchemy, can be considered as a useful resource for topic extraction. They also show that a substantial increase in recall can be achieved by combining the annotators with a weighted voting scheme. Finally, an interesting result is that by removing Alchemy from the combination, or by combining only the more precise annotators, we get a significant increase in precision, at the cost of a lower recall.

Categories and Subject Descriptors

[**Natural Language Processing**]: [Text analysis, Information extraction]

Keywords

Semantic annotation, topic extraction, evaluation

1. INTRODUCTION

Semantic annotation is the process of identifying expressions in texts and linking them to some entities in a knowledge base. The importance of semantic annotation for the Semantic Web is not anymore to be demonstrated. The Semantic Web realization depends on the availability of metadata, defined through some formal semantic structures, and describing web content. Thus, the acquisition of metadata through the development of automatic annotation tools is a major challenge for the Semantic Web. There have been many semantic annotators developed based on ontologies [1, 2] during the past decade, and some comparative studies have been performed on these annotators [3, 4]. However, a recent trend has emerged with the development of semantic annotators that are based on the Linked Open Data cloud (LOD) [5]. Typically, they detect named entities and concepts and link them to some semantic entities in a given LOD dataset. The most prominent and exploited dataset is DBpedia [6], which is considered as a hub on the LOD.

Due to the recent emergence of linked data annotators, very few comparative studies have been published on their performance [7, 8]. The available studies generally focused on traditional named entities (e.g. person, organization, etc.) in their evaluation. These studies were mainly interested in evaluating the capability of the annotators to correctly link expressions to semantic entities. Recently, the Knowledge Base Population track has proposed a specific task named *entity-linking* dedicated to the evaluation of such annotators. In that context, annotators have to link a named entity mention in a document with its corresponding entry in Wikipedia. [9, 10] give an overview of the methods used by the track participants. Previous studies did not draw any conclusion about the relevance of the detected entities, which is a crucial point. We know that semantic annotators have not necessarily been developed with the objective of finding only expressions that are relevant to the domain. Nevertheless, one might be interested in an annotator that does not only correctly disambiguate entities, but also produce a relevant set of annotations. Evaluating the existing annotators in this respect will help us determine how far we are from a state where they could be efficiently used for such a task.

In this paper, we address directly this issue. Our main research questions can be articulated around the following points: i) whether linked data-based semantic annotators can be used to extract relevant expressions in a **specific domain** and ii) whether the performance of individual semantic annotators can be improved using voting methods and machine learning. Answering these two research questions is crucial for the further development and use of semantic annotators and is of major importance to several communities, including the text mining and the Semantic Web community. Noting that there are two kinds of annotated expressions, that is, named entities and other expressions that are linked to conceptual entities (which we will call *topics* in this paper), we also evaluated the relevance of detected expressions separately for these two kinds of annotations.

To answer these research questions, we evaluate the performance of seven state-of-the-art linked data annotators,

WWW 2013 Companion, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2038-2/13/05.

namely DBPedia Spotlight[1][11], Wikimeta[2], OpenCalais [3], Alchemy [4], Lupedia [5], Yahoo Content Analysis API [6] and Zemanta [7]. Section 2 details each annotator. Then section 3 describes the research methodology used to perform the annotations, create a domain-dependent Gold Standard, and compare the results. In section 4, we show the performances for each individual annotator. In section 5 we describe the voting methods and machine-learning algorithms used in our experiments, followed by the presentation of their performances, in section 6.

2. SEMANTIC ANNOTATORS

As previously said, semantic annotation, also called entity linking, consists in identifying a unique and precise link between a word sequence and a semantic resource descriptor, among those available in repositories such as the Linked Open Data (LOD) datasets. Semantic annotation is based on various techniques in natural language processing (NLP) and machine learning for entity extraction and relies upon Semantic Web knowledge representations such as ontologies and open linked data for entity linking (see, for example, [11, 12, 13, 14]).

Semantic annotation can be seen as encompassing the named entity recognition task, which limits itself to the identification of expressions belonging to a closed set of class labels, such as person, product, organization, etc. For example, in the sentence *Barack Obama is the President of the United States*, the NER task would identify *Barack Obama* as a person and *United States* as a geographical location. In contrast, semantic annotation would associate these two expressions to a unique entity in a knowledge base such as DBpedia (`http://dbpedia.org/resource/Barack_Obama` and `http://dbpedia.org/resource/United_States`, respectively).

The following sections give a short description of the most prominent semantic annotators used in this evaluation. In some cases, the description is very generic due to the lack of published descriptions of the services.

AlchemyAPI employs sophisticated deep linguistic parsing and statistical language processing for performing the annotations. It offers various APIs, among which two are relevant for our experiments: named entity extraction and keyword extraction. The named entity extractor is able to disambiguate the detected entities and resolve co-references. Entities are linked to various datasets on the LOD. Keyword extraction focuses on topics, but cannot be considered as an "annotation task" per se, as the service produces a list of keywords and does not indicate the portions of texts that refer to these keywords. By default, the keyword extractor returns a maximum of 50 keywords. Contrarily to keywords, the position of named entities in texts is returned by the named entity extractor.

[1]`https://github.com/dbpedia-spotlight/dbpedia-spotlight/`
[2]`http://wikimeta.org/`
[3]`http://http://www.opencalais.com/`
[4]`http://www.alchemyapi.com/`
[5]`http://lupedia.ontotext.com/`
[6]`http://developer.yahoo.com/contentanalysis/`
[7]`http://developer.zemanta.com/`

DBpedia Spotlight achieves semantic annotation using a three-step approach [11]. The first step, the spotting phase, is the identification of candidate word sequences that could be linked to an entity in the DBpedia dataset. DBpedia Spotlight offers various methods (spotters) for this phase. Then for each candidate word sequence in text, DBpedia Spotlight pre-ranks DBpedia entities for which there is an associated label that corresponds to the sequence surface form. Finally, in the disambiguation phase, DBpedia Spotlight uses a similarity score to determine which candidate entity is the most relevant. The similarity score takes into account the context of the expression (a window of words around the expression) and the context of each candidate entities. Since each DBpedia entity is represented as a URI that mirrors a corresponding entry in Wikipedia, the union of the set of words around the Wikipedia hyperlinks that point to the corresponding URI is used as a context for these candidate entities.

Wikimeta performs named entity detection as a first step using a statistical model. It then tries to link each detected named entity to some entity in DBpedia based on a disambiguation process that is described in [12]. Similarly to DBpedia Spotlight, a word context around the expression is compared to the contexts of candidate resources in DBpedia. Essentially, Wikimeta differs from DBpedia Spotlight in the way expressions are detected in texts and in the lexical resource structure used to find candidate semantic entities.

Yahoo! Content Analysis API detects several types of expressions including entities, concepts, categories, and relationships within texts. Entities are ranked by their overall relevance, and some of these entities are then mapped to Wikipedia pages when possible. Unfortunately, very few details regarding the techniques used for the implementation of the service are available.

Open Calais semantic annotations include entities, facts, events and categories. Expressions in texts are linked to entities described in Open Calais ontology rather than on the LOD . However, some entity types, such as cities, countries or companies can be further described with a link to DBpedia. Open Calais also relies on natural language processing (NLP), machine learning and other methods to create its annotations.

Lupedia enrichment service is mainly a named entity recognition system. It uses a gazetteer, which is essentially a list of surface forms that are associated to a subset of entities in DBpedia (events, organizations, persons, places and works (e.g. musical or artistic)) and LinkedMDB (a dataset that contains movies descriptions: films, directors, actors, etc.). The default configuration takes the longest sequence of consecutives words that corresponds to some entry in the gazetteer and annotates it with the corresponding entity in the knowledge base. The annotation does not take into consideration the context of the expression to disambiguate between multiple potential candidates, contrary to Wikimeta or DBPedia Spotlight.

Zemanta: By opposition to most of the other semantic annotators, Zemanta does not annotate the position of each detected expression in texts. Rather, it provides a set of expressions, which are not necessarily found in texts and it identifies topics that can represent the content as a whole. For each expression, Zemanta specifies a list of links that point to corresponding entries in some knowledge resources, such as Wikipedia, IMDB, MusicBrainz and Amazon book listings.

Table 1 summarizes the main characteristics of these semantic annotators.

Annotator	Detects	Pos. in text
Alchemy	NE/Top	Only for NE
Spotlight	NE/Top	yes
Wikimeta	Mainly NE	yes
Lupedia	NE	yes
Open Calais	NE/Top	yes
Yahoo	NE/Top	yes
Zemanta	Top	no

Table 1: Characteristics of semantic annotators (NE = named entites, Top = Topics)

3. RESEARCH METHODOLOGY

To understand to what extent semantic annotators are adequate for the extraction of relevant domain expressions, we made two experiments. First, we evaluated the performances of the semantic annotators separately on a domain corpus. As we will see, most of the annotators are not sufficiently efficient in terms of precision and recall to identify domain relevant expressions. In our second experiment, various combination approaches have been tested. In all experiments, an expression is considered relevant if it is directly related to the content of the document where it appears. Thus, by taking the union of the sets of relevant expressions extracted from each document, we obtain a set of expressions that may be considered as relevant for the domain.

In our experiments, we analyzed the performance in three different situations, depending on the type of expressions that are considered: all expressions, only named entities and only topics (that is, every relevant expression that is not a named entity). To perform our experiments, we relied on a corpus of 8 texts taken from Wikipedia, all related to the artificial intelligence domain. Together, these texts represent 10570 words. By applying all annotators to these texts, we obtained 2023 expression occurrences among which 1151 were distinct.

We asked a human evaluator to analyze each expression and make the following decisions:

- Does the annotated expression represent an understandable named entity or topic? To be understandable, an expression must be a complete and well-formed expression (not part of another expression) and it must be semantically significant. Verbs, adjectives, adverbs and generic nouns (like *person*, *system*, *theory*) are not considered as understandable annotations.
- Is the expression a relevant keyword according to document content?
- Is the expression a relevant named entity according to document content?

The last question is not simple, because the notion of named entity has evolved with the latest advancements of semantic annotators. Traditional named entity definition involves entities with common names such as persons, locations, organizations, products, events and dates. However, we have witnessed a tendency to refine and extend possible categories of named entities through various taxonomies defined for semantic annotators. The NERD evaluation platform [8], in an attempt to unify these taxonomies, describes 85 possible categories, including sport events, operating systems, political events and websites, all of which cannot be considered as "traditional" named entities. Similarly, in specialized domains, such as the biomedical domain, a specific fine-grained categorization of named entities is used and includes genes, proteins, diseases, drugs, or organisms [15]. In the computer science domain, some questions regarding the possible categorizations emerge as well, as we must decide if some expressions such as XML and RDF are named entities or topics. For example, XML and RDF somehow refer to unique instances of entities (here metadata languages), but their use in sentences such as *An RDF-based data model is naturally suited to certain kinds of knowledge representation* does not reflect the usual definition of a named entity. In our evaluation, we adopt the "classical interpretation" of named entities and consider, in this case, XML and RDF as topics.

The answers of the human annotator allowed us to build a Gold Standard whose expression distribution is given in Table 2. Note that only 639 out of 1151 detected expressions are understandable (56%). Thus a substantial number of detected expressions represents noise. However, a high ratio of understandable expressions is considered relevant (79%). It is also interesting to note that, while 11% of detected expressions are named entities, few of them are relevant. In fact, only 6% of relevant expressions are named entities (30 out of 507).

Type of expression	Quantity	Nb. Relevant
Total detected	1151	
Total understandable	639	507
Topics	516	477
Named entities	123	30

Table 2: Distribution of expressions in our Gold Standard. We consider only distinct expressions.

Table 3 shows all expressions detected at least 6 times in our corpus. For each expression, we also indicate whether it has been tagged as understandable and relevant.

4. PERFORMANCES OF SEMANTIC ANNOTATORS

Using the obtained Gold Standard, we evaluated the output of each semantic annotator using standard information retrieval measures: precision, recall and F-measure. Precision, recall and F-score are defined in the following way:
Let

$S_A =$	$a_i, \dots a_n$ where a_i is an annotator.
$N_D(a_i) =$	Set of distinct expressions detected by the annotator a_i.
$N_R(a_i) =$	Set of distinct relevant expressions detected by the annotator a_i.
$N_O =$	$\bigcup_{a_i \in S_A} N_R(a_i)$.

Expression	# occ.	Und.	Rel.
artificial intelligence	38	yes	yes
intelligence	18	yes	yes
AI	15	yes	yes
intelligent agent	13	yes	yes
unknown	12	no	no
machine learning	12	yes	yes
Arthur C. Clarke	12	yes	no
John McCarthy	11	yes	yes
data mining	11	yes	yes
computer science	11	yes	yes
Ray Kurzweil	10	yes	yes
knowledge	10	yes	yes
intelligent agents	9	yes	yes
Thomas Nagel	8	yes	no
system	8	no	no
Russell	8	no	no
neural network	8	yes	yes
natural language processing	8	yes	yes
Marvin Minsky	7	yes	yes
Hubert Dreyfus	7	yes	no
computational linguistics	7	yes	yes
world	6	no	no
Web Ontology Language	6	yes	yes
theory	6	no	no
systems	6	no	no
SPSS	6	yes	yes
software engineering	6	yes	yes
semantic web	6	yes	yes
science	6	yes	yes
Roger Penrose	6	yes	no
learning	6	yes	yes
Hubble space telescope	6	yes	no
Herbert Simon	6	yes	yes
FIPA	6	yes	yes
algorithms	6	yes	yes

Table 3: List of expressions detected at least 6 times. For each expression, we indicate if it has been tagged as understandable and relevant.

$$\text{Prec}(a_i) = \frac{|N_R(a_i)|}{|N_D(a_i)|}$$

$$\text{Rec}(a_i) = \frac{|N_R(a_i)|}{|N_O|}$$

$$\text{F-Score}(a_i) = \frac{2\times\text{Prec}(a_i)\times\text{Rec}(a_i)}{\text{Prec}(a_i)+\text{Rec}(a_i)}$$

Note that we adopted an unusual way of computing recall, since our Gold Standard has been produced by considering only the expressions detected by at least one annotator, instead of considering independently all expressions that are found in our corpus. To obtain values that are comparable to the ones obtained using combination methods (see section 5), precision and recall values are not computed on the whole extracted expressions, but rather averaged over five balanced partitions of the expressions[8].

Table 4 shows the results obtained for each annotator taken individually. We can notice that F-score is low for almost all annotators. This is not really surprising, considering that semantic annotators usually do not have as objective the extraction of relevant expressions. Low precision of Wikimeta is explained by the fact that it detected some named entities, like times and dates, that are not usually relevant as keyphrases. DBpedia Spotlight annotated many expressions that are not considered understandable. Note that if we consider only understandable expressions, precision values for Alchemy, Spotlight and Wikimeta would be 0.81, 0.73 and 0.74, respectively, Lupedia's precision would still be low (0.43) and, as expected, precision values for Open Calais, Yahoo and Zemanta would remain very high (0.72, 0.87 and 0.88, respectively).

[8]Exactly the same partitions used for evaluating combination methods.

All expressions						
Method	Det	Top	NE	Und.	Rel	P/R/F
Alchemy	619	564	55	428	347	0.56/0.69/0.62
Spotlight	356	338	18	149	108	0.3/0.21/0.25
Wikimeta	243	149	94	130	96	0.39/0.19/0.25
Lupedia	42	21	21	28	12	0.27/0.024/0.043
Open Calais	147	107	40	127	91	0.62/0.18/0.28
Yahoo	105	93	12	85	74	0.7/0.15/0.24
Zemanta	77	69	8	69	61	0.77/0.12/0.21
Only named entities						
Alchemy	55	0	55	50	18	0.37/0.62/0.44
Spotlight	18	0	18	17	8	0.44/0.27/0.32
Wikimeta	94	0	94	51	24	0.25/0.78/0.38
Lupedia	21	0	21	19	6	0.31/0.22/0.24
Open Calais	40	0	40	39	12	0.33/0.39/0.35
Yahoo	12	0	12	11	5	0.28/0.15/0.2
Zemanta	8	0	8	8	5	0.47/0.17/0.25
Only topics						
Alchemy	564	564	0	378	329	0.59/0.69/0.63
Spotlight	338	338	0	132	100	0.3/0.21/0.25
Wikimeta	149	149	0	79	72	0.48/0.15/0.23
Lupedia	21	21	0	9	6	0.3/0.012/0.023
Open Calais	107	107	0	88	79	0.75/0.17/0.27
Yahoo	93	93	0	74	69	0.74/0.15/0.24
Zemanta	69	69	0	61	56	0.81/0.12/0.21

Table 4: Results obtained for each annotator taken individually: number of detected expressions(Det), topics (Top), named entities (NE), understandable (Und.), relevant (Rel), precision (P), recall (R) and F-score (F).

One semantic annotator that clearly distinguishes itself is Alchemy, which obtains a very high value for recall when all entities or only topics are considered. This means that a great proportion of relevant expressions detected by other annotators are also detected by Alchemy. But the precision of Alchemy's results is not very high, compared to the results of Open Calais, Yahoo and Zemanta, which exhibit the highest precision. However, recall values for these three annotators are very low. Interestingly, peformances are very low when only named entities are considered: none of the annotators was good at precisely detecting relevant named entities. Note the high recall value obtained with Wikimeta.

Table 5 describes the frequency of detected expressions. As can be noticed, most of the expressions have been detected by only one (76%) or two (16%) annotators. From these results, we can conclude that semantic annotators are complementary. Thus it is reasonable to expect that a combination of their decisions would provide better performances.

5. COMBINATION METHODS

To test the above-mentioned hypothesis, we implemented a number of voting measures. An evident combination method is a vote among the annotators. In its simplest implementation, an expression is considered relevant if a minimum of m among n annotators detect it. Another strategy is to produce a weighted vote by using the precision of each indi-

N	Number of expressions
1	872
2	183
3	57
4	24
5	8
6	5
7	2
Total	1151

Table 5: Frequency of detected expressions. For each value N, we provide the number of distinct expressions that have been detected by exactly N annotators.

vidual annotator on a training corpus. Given the set of annotators used in the experiment $S_A = \{a_1 \dots a_n\}$, $ann(i,e)$ is equal to 1 if expression e is detected by a_i and 0 otherwise. More formally, the voting methods may be described in the following way:

Simple vote: For each detected expression e, return e as a relevant expression if $\sum_{a_i \in S} ann(i,e) \geq th$, where th is a predefined threshold, such that $0 \leq th \leq n$.

Weighted vote: For each detected expression e, return e as a relevant expression if $\sum_{a_i \in S} w_i \times ann(i,e) \geq th$, where:

th is a predefined threshold, such that $0 \leq th \leq 1$,

$prec(a_i)$ denotes the precision for annotator a_i,

and weight w_i is defined as $\frac{prec(a_i)}{\sum_{a_i \in S} prec(a_i)}$.

As can be noticed, the precision of an annotator is obtained by computing the ratio of relevant expressions among the ones detected by each annotator. Relevant expressions are those defined in the Gold Standard.

Another voting scheme is the K-nearest-neighbours classifier. For each detected expression e in a training dataset, we define a vector $v(e)$ composed of the n decisions of annotators a_1 to a_n for this expression. Given D, the set of detected expressions in the training dataset, we obtain a set $T = \{\langle v(e), rel(e)\rangle \mid e \in D\}$, where $rel(e) = 1$ if e is a relevant expression. To classify a new expression e, we simply find the K closest vectors $T_K \subseteq T$ and consider e as relevant if it is relevant for a majority of vectors in T_K.

Finally, we also experimented with various classical machine learning methods. For all these methods, we used Weka's implementations with default configurations: Naive Bayes classifier, decision tree and rule induction. For decision tree, we selected the C4.5 algorithm, and for rule induction, the PART algorithm [16], where rules are extracted from partial decision trees.

6. EVALUATION OF COMBINATION METHODS

To evaluate the combination methods, we partitioned the set of detected expressions into 5 balanced partitions. By selecting one partition for testing and the remaining ones for training, we realized 5 experiments for each of the following situations: all entities, only named entities and only topics.

For each combination method (weighted vote, K-nearest neighbors, Naive Bayes, decision tree and rule induction), a relevant expression classifier is obtained based on the training set. Then, the test set is used to compute precision, recall and F-measure. Precision for combination methods is the ratio of expressions correctly classified as relevant expressions, whereas recall is the ratio of the entire set of relevant expressions (according to the Gold Standard) that has been recognized by the classifier. As we said earlier, we adopted an unusual way of computing recall, since our Gold Standard is produced by considering only the expressions detected by annotators, instead of considering independently all possible relevant expressions in the corpus (some relevant expressions might have been missed by all annotators).

More formally, let D be the set of detected expressions in our corpus. S_g is the set $\{e \mid e \in D, a_i \in S_A, \exists a_i$ such that $ann(i,e) = 1, rel(e) = 1\}$. Put simply, S_g is the set of relevant expressions that have been detected by at least one annotator. Let $Dec(e)$ be the decision made by our classifier for expression e (returns 1 if it classifies e as a relevant expression, and 0 otherwise). $S_e = \{e \mid e \in D, Dec(e) = 1\}$ is the set of expressions considered as relevant by our classifier and $S_r \subseteq S_e$ is the subset of these expressions that are relevant expressions according to Gold Standard, that is, $S_r = \{e \mid e \in S_e, rel(e) = 1\}$. Precision, recall and F-score are defined in the following way:

Precision: $\mathrm{P} = \frac{|S_r|}{|S_e|}$

Recall: $\mathrm{R} = \frac{|S_r|}{|S_g|}$

F-score: $\mathrm{F} = \frac{2 \times P \times R}{P + R}$

For all the voting methods (simple vote and weighted vote), a threshold must be determined. Table 6 provides the performance values under various thresholds for the simple vote method. The highest F-score is obtained with threshold = 1, but it is due to the fact that all expressions are classified as relevant expressions, since all of them are detected by at least one annotator. Therefore, it is not surprising that none of the actual relevant expressions has been missed. However, precision is too low. In fact, we may consider this situation as our baseline. The optimal value for threshold is 2, but still, this value does not outperform any of the best annotators taken individually.

Table 7 reports the performance values obtained with the weighted vote method. We can observe that a threshold of 0.10 gives the best F-score value, for cases where all entities or all topics are considered. For named entities, the threshold must be slightly higher to obtain good performances. If we consider that precision is more important than recall, a threshold of 0.15 would be a better choice. We also see that the F-score value is higher than the value obtained by annotators taken separately (for Alchemy the increase may be not significant).

If precision is very important, Table 7 shows that the weighted-vote approach may be a good solution if we use a threshold of 0.25, in which case the precision is 0.69. Recall is low, but we still obtain 45 detected expressions, on average. In this case, the list of extracted expressions for one

Threshold	P	R	F
All expressions considered			
1	0.44	1.0	0.61
2	0.65	0.36	0.46
3	0.67	0.13	0.21
4	0.6	0.047	0.086
5	0.55	0.018	0.034
6	0.4	0.0098	0.019
7	0.1	0.0019	0.0036
Only named entities considered			
1	0.24	1.0	0.39
2	0.38	0.62	0.47
3	0.45	0.5	0.46
4	0.34	0.26	0.29
5	0.3	0.095	0.14
6	0.4	0.095	0.15
7	0.2	0.029	0.05
Only topics considered			
1	0.46	1.0	0.63
2	0.72	0.34	0.46
3	0.83	0.1	0.18
4	0.9	0.036	0.069
5	0.8	0.012	0.024
6	0.2	0.0043	0.0083
7	0.0	0.0	0.0

Table 6: Precision (P), recall (R) and F-score (F) values obtained for simple-vote method.

partition would be the following one[9]: *RDF Schema, (Hubert Dreyfus), Computational learning theory, (Roger penrose), (lengthy research), science, (AGIRI), Semantic Web, Neural network, Stock market analysis, bioinformatics, continuous planning algorithm, (Japan), DAG, graphical model, research, (computer hardware), (human traits), (John), pattern recognition, semantic networks, (chess), (costly search processes), NLP, XML, NLP algorithms, weak AI hypothesis, Dijkstra, artificial neural network, ILP, (data mining efforts), (Oxford University Press), (scientist), singularity institute for artificial intelligence, (human), Norvig, machine learning algorithms, (William Clocksin), hierarchical task networks, SVM.*

Tables 8 compares the results of the machine learning approaches and the best voting method, which is weighted vote. On the average, weighted vote displays the best precision among combination methods if only topics are considered. In this case, we see that precision obtained with weighted vote is significantly higher than the one obtained with Alchemy (0.66 vs 0.59). Machine learning methods achieve better recall and F-score on the average, especially if decision tree or rule induction (PART) is used. Finally, we see that considering only named entities, the combination methods do not improve the performances.

Finally, Table 9 compared the performances obtained with weighted vote using all annotators with a combination that excludes Alchemy. We notice a slight increase in average precision, with a maximum value of 0.78. This suggests that Alchemy has the effect of finding many relevant expressions that are not discovered by other annotators (recall of 0.78 vs 0.37), but at the cost of some noise. Also, the negative effect on precision due to the presence of Alchemy may be explained by the fact that it detects substantially more expressions than any other annotator. Thus, the probability of adding its vote to an expression is high, with the effect of pushing the score over the threshold for many irrelevant expressions that would otherwise bien overlooked. More detailed analysis must be achieved to obtain a clear explanation of this phenomenon.

[9]Irrelevant expressions, according to our Gold Standard, are enclosed within parenthesis.

Threshold	P	R	F
All expressions considered			
0.0	0.44	1.0	0.61
0.05	0.44	1.0	0.61
0.1	0.52	0.92	0.66
0.15	0.57	0.78	0.64
0.2	0.67	0.39	0.49
0.25	0.69	0.31	0.42
0.3	0.7	0.24	0.36
0.35	0.69	0.16	0.26
Only named entities considered			
0.0	0.24	1.0	0.39
0.05	0.24	1.0	0.39
0.1	0.26	0.85	0.4
0.15	0.36	0.7	0.48
0.2	0.38	0.65	0.48
0.25	0.47	0.57	0.5
0.3	0.46	0.54	0.49
0.35	0.39	0.35	0.36
Only topics considered			
0.0	0.46	1.0	0.63
0.05	0.46	1.0	0.63
0.1	0.56	0.91	0.69
0.15	0.66	0.53	0.56
0.2	0.7	0.37	0.48
0.25	0.78	0.29	0.42
0.3	0.8	0.23	0.35
0.35	0.88	0.12	0.21

Table 7: Average values for precision (P), recall (R) and F-score (F) with weighted-vote method.

We also produced one additional combination, by taking into account only the most precise (on an individual basis) annotators OpenCalais, Yahoo and Zemanta. As expected, this had the effect of increasing the precision (0.64) while also improving the recall of each of these annotators separately.

7. CONCLUSIONS

In this paper, we presented a first attempt to evaluate the capability of linked data semantic annotators to detect expressions that are relevant to a domain. We are aware that these annotators were not necessarily developed with this objective (especially DBpedia Spotlight and Wikimeta), but this does not undermine the pertinence of this kind of evaluation. Our results show clearly that if relevancy to the domain is important, the extraction of key expressions achieved by these tools would be only a first step that should be followed by some filtering or refinement. Even if we consider only named entities, for which these annotators are

Method	Average P/R/F	Min P/R/F	Max P/R/F
All entities considered			
Weighted vote (th = 0.15)	0.57/0.78/0.64	0.53/0.43/0.49	0.59/0.9/0.71
KNN (N=3)	0.58/0.71/0.62	0.54/0.39/0.48	0.62/0.86/0.68
Bayes	0.57/0.72/0.63	0.52/0.68/0.62	0.58/0.77/0.65
Dec. tree	0.58/0.82/0.67	0.54/0.78/0.66	0.6/0.87/0.7
PART	0.57/0.82/0.68	0.54/0.78/0.66	0.59/0.87/0.7
Only named entities considered			
Weighted vote (th = 0.15)	0.36/0.7/0.48	0.2/0.43/0.27	0.54/1.0/0.7
KNN (N=3)	0.48/0.12/0.17	0.0/0.0/0.0	1.0/0.29/0.33
Bayes	0.35/0.15/0.19	0.0/0.0/0.0	1.0/0.33/0.5
Dec. tree	0.04/0.05/0.044	0.0/0.0/0.0	0.2/0.25/0.22
PART	0.47/0.095/0.15	0.0/0.0/0.0	1.0/0.17/0.29
Only topics considered			
Weighted vote (th = 0.15)	0.66/0.53/0.56	0.53/0.39/0.51	0.73/0.89/0.66
KNN (N=3)	0.59/0.73/0.64	0.52/0.43/0.52	0.66/0.84/0.7
Bayes	0.59/0.77/0.67	0.53/0.73/0.63	0.66/0.82/0.73
Dec. tree	0.59/0.83/0.69	0.54/0.73/0.66	0.64/0.9/0.75
PART	0.59/0.83/0.69	0.54/0.72/0.66	0.64/0.9/0.75

Table 8: Average, min and max values obtained for precision (P), recall (R) and F-score (F).

Method	Average P/R/F	Min P/R/F	Max P/R/F
All annotators	0.57/0.78/0.64	0.53/0.43/0.49	0.59/0.9/0.71
Without Alchemy	0.63/0.37/0.47	0.54/0.29/0.38	0.78/0.46/0.58
Only Open-Calais, Yahoo and Zemanta	0.64/0.35/0.45	0.54/0.24/0.33	0.79/0.44/0.57

Table 9: Peformance, using weighted vote with threshold = 0.15

usually good, performance would not be sufficient: few of the detected named entities are relevant.

Another observation we made is that semantic annotators are complementary: most of the expressions have been detected by at most two annotators. This led us to experiment combination methods, which revealed a clear improvement on recall, meaning that more relevant expressions are detected. But precision remains unsatisfactory, at about 0.57. Our results show that if we could automatically identify named entities and exclude them from the set of detected expressions, precision would increase significantly up to 0.66 if we used weighted vote, but at the cost of a lower recall (0.53).

It is important to note that in our combination experiments, the only feature we use is whether the annotator has detected an expression or not. But there are more features that could be used: some annotators try to identify the category of the entity and some provide a link to Wikipedia, DBpedia or some other resource. We expect that using these features would help increase the precision.

The kind of evaluation we tried to realize is made difficult by the absence of evaluation corpora. We had to create our own evaluation corpus, which is costly. We are aware that our expriment should be repeated with a larger dataset, previously annotated such that real recall values could be computed, but nevertheless our results already help us to make a useful assessment on the capability of semantic annotators to detect relevant entities. We are planning to work on the creation of such a large corpus, which will be made available to the research community.

In future works, apart from taking into consideration more features in combination methods, we will explore other annotators. We will also compare the results of linked data semantic annotators with keyphrase extractors.

8. REFERENCES

[1] Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A., Tomlin, J., Zien, J.: Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation. In: Proceedings of the 12th International Conference on World Wide Web, Budapest, Hungary (2003) 178–186

[2] Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., Goranov, M.: KIM - semantic annotation platform. In: 2nd International Semantic Web Conference (ISWC 20013), Sanibel Island, Florida (2004) 834–849

[3] Reeve, L., Han, H.: Survey of semantic annotation platforms. In: Proceedings of the 2005 ACM symposium on Applied computing. SAC '05, New York, NY, USA, ACM (2005) 1634–1638

[4] Rajput, Q., Haider, S.: A comparison of two ontology-based semantic annotation frameworks. In: AIAI. (2010) 187–194

[5] Heath, T., Bizer, C.: Linked data: Evolving the web into a global data space. Synthesis Lectures on the Semantic Web: Theory and Technology **1** (2011) 1–136

[6] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, C., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: Proceedings of the 6th International Semantic Web Conference (ISWC2007), Busan, Korea (2007)

[7] Hachey, B., Radford, W., Nothman, J., Honnibal, M., Curran, J.R.: Evaluating entity linking with wikipedia. Artificial Intelligence **194** (2013) 130–150

[8] Rizzo, G., Troncy, R., Hellmann, S., Bruemmer, M.: NERD meets NIF: Lifting NLP extraction results to the linked data cloud. In: Linked Data on the Web (LDOW2012). (2012)

[9] Ji, H., Grishman, R., Dang, H.T.: Overview of the TAC 2011 knowledge base population track. In: Proceedings of the Text Analysis Conference. (2011)

[10] Ji, H., Grishman, R.: Knowledge base population: Successful approaches and challenges. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics (2011)

[11] Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: shedding light on the web of documents. In: Proceedings of the 7th International Conference on Semantic Systems. I-Semantics '11, New York, NY, USA, ACM (2011) 1–8

[12] Charton, E., Gagnon, M., Ozell, B.: Automatic semantic web annotation of named entities. In: Canadian Conference on AI. (2011) 74–85

[13] Oren, E., Hinnerk Möller, K., Scerri, S., Handschuh, S., Sintek, M.: What are semantic annotations? Technical report, DERI (2006)

[14] Grassi, M., Morbidoni, C., Nucci, M., Fonda, S., Ledda, G.: Pundit: Semantically structured annotations for web contents and digital libraries. In: Proceedings of the Second International Workshop on Semantic Digital Archives (SDA 2012), Paphos, Cyprus (2012)

[15] Leaman, R., Gonzalez, G.: BANNER: An executable survey of advances in biomedical named entity recognition. In: Pacific Symposium on Biocomputing. (2008) 652–663

[16] Holmes, G., Mark, H., Eibe, F.: Generating rule sets form model trees. In: Australian Joint Conference on Artificial Intelligence. (1999) 1–12