Semantic Annotation of Quantitative Textual Content

Mehrnaz Ghashghaei[1,2], John Cuzzola[2], Ebrahim Bagheri[2], Ali A. Ghorbani[1]

[1] University of New Brunswick
[2] Laboratory for Systems, Software and Semantics (LS^3), Ryerson University

Abstract. Semantic annotation techniques provide the basis for linking textual content with concepts in well grounded knowledge bases. In spite of their many application areas, current semantic annotation systems have some limitations. One of the prominent limitations of such systems is that none of the existing semantic annotator systems are able to identify and disambiguate quantitative (numerical) content. In textual documents such as Web pages, specially technical contents, there are many quantitative information such as product specifications that need to be semantically qualified. In this paper, we propose an approach for annotating quantitative values in short textual content. In our approach, we identify numeric values in the text and link them to an existing property in a knowledge base. Based on this mapping, we are then able to find the concept that the property is associated with; whereby, identifying both the concept and the specific property of that concept that the numeric value belongs to. Our experiments show that our proposed approach is able to reach an accuracy of over 70% for semantically annotating quantitative content.

1 Introduction

As more and more content is being disseminated on online platforms such as blogs, social media and microblogs, the need for better and more efficient techniques for organizing, searching and efficiently retrieving information is required. Techniques that benefit from well-grounded knowledge bases such as ontologies for the sake of information organization and retrieval have received attention in the recent years [7], which include open information extraction [8], ontology population and enrichment [9], and semantic tagging and annotation [1,2], just to name a few. These techniques aim to identify and extract structured information from unstructured content. Automated semantic annotation systems are among such systems that enable the identification and labeling of instances of knowledge base concepts within text; whereby, enriching textual documents with additional semantic information linked to external knowledge bases.

With the emergence of the linked open data initiative, many semantic annotator systems now benefit from the knowledge bases that are shared through this platform to spot, disambiguate and link semantic information within textual content. Knowledge bases such as Freebase and DBpedia that sit at the core of the linked open data cloud have been used extensively for this purpose where their concepts are employed for semantically grounding textual content. Semantic annotator systems typically provide support for entity linking, suggestion of related but unobserved concepts, role assignment and detection of relevant semantic categories.

In spite of the growing adoption of semantic annotator systems, one of the major limitations that current annotators face concerns dealing with quantitative (numerical) textual content. In other words, none of the existing semantic annotator systems is able to semantically link or describe numerical content. Therefore, valuable information that are expressed in the form of numbers are largely ignored in the current semantic annotator systems; hence, they are neither exploited in the annotation process nor are they semantically linked for future use.

Let us consider a sample short text describing a Samsung Galaxy S smart phone: "The Samsung Galaxy S uses the Samsung S5PC110 processor. This processor combines a 1 GHz ARM Cortex-A8 based CPU core with a PowerVR SGX 540 GPU made by Imagination Technologies.". When processed by a state of the art semantic annotator system such as TagMe [10], the phrases 'Samsung Galaxy S', 'ARM Cortex-A8', 'processor', 'Imagination Technologies' and 'CPU' are detected and linked to their corresponding Wikipedia entities. However, none of the numerical values are detected for semantic annotation. This limitation prevents the correct interpretation of quantitative values within text, which can constitute a noticeable portion of text, e.g. see product specification Web pages.

In this paper, we propose an approach for annotating quantitative values in a short text. In our work, we identify numeric values in text and not only link them to the most relevant property in the knowledge base but also find the best matching concept[1] that has the identified property. Therefore, our method enables the specification of a numeric value within the context of a concept and by relating it with one of the properties of that concept. For instance, in the above example, our method is able to determine that 1 GHz is the value of the *frequency* property of the *ARM Cortex-A8* concept.

For evaluating our approach, we exploit a gold standard dataset consisting of short textual snippets that have at least one numerical value. We compare the obtained property and concept for each numerical value and compare them with the gold standard. The results of our evaluation show that our method is able to correctly identify the most relevant concept and corresponding property in over 70% of the cases.

The rest of this paper is organized as follows. In Section 2, we review the background on automated semantic annotation of textual content. Section 3 is a detailed description of our proposed approach including the procedure for identifying the relevant entities and corresponding properties. The evaluation procedure, dataset and results are provided in Section 4 and finally Section 5 concludes the paper.

[1] Also known as *entity*.

2 Background

One of the open areas of knowledge extraction from natural language is semantic annotation. For the sake of brevity, we refer to semantic annotation tools as annotators. Annotation is basically the task of extraction and disambiguation of mentioned entities in a given text. Annotators typically operate based on three main phases: detection of concept candidates, disambiguation, and pruning of results [1], which we briefly review in the following.

2.1 Detection

In the first phase, the annotator processes the given input text and picks out specific phrases from the text, called "mentions", that can potentially refer to an existing concept with the source knowledge base. For each of the mentions, a set of candidate concepts are selected that are associated with that mention. Detection of mentions is also known as "spotting". TagMe [10] has an Anchor Dictionary for this phase, and detects mentions by querying this dictionary. DBpedia Spotlight [3] also relies on a dictionary for spotting. In DBpedia Spotlight, a lexicon that associates multiple surface forms to a concept is used. Wikipedia Miner [4] uses pure text processing to find the spots and their candidates. It gathers all n-grams within text but only keeps those that have a high probability of linking in order to discard irrelevant phrases and stop words. In AIDA [5], a Named Entity Recognition (NER) tool is used. This NER tool identifies noun phrases that potentially denote named entities. Then YAGO2 is used to associate a candidate set to each potential named entity. In Illinois Wikifier [6] the authors perform pure text processing for entity spotting. They utilize an anchor-title index, computed by crawling Wikipedia, that maps each distinct hyperlink anchor text to its target Wikipedia titles. Since checking all substrings in the input text against the index is computationally inefficient, they only consider the expressions marked as named entities by a NER tagger, the noun-phrase chunks extracted by a publicly available shallow parser, and all sub-expressions of up to 5 tokens of the noun-phrase chunks. Then, for each mention, Wikipedia titles that are mapped to the mention (anchor text) are considered to be the candidate entities.

In our work, the detection phase starts with finding the numeric values in the input text. Assuming that we have the disambiguated mentions in the text, a set of candidate concepts are extracted. These concepts have the potential of having the most relevant property for the numeric value. Then from all properties of candidate concepts, a set of candidate properties are selected and associated with the spotted numeric value.

2.2 Disambiguation

Within the detection phase, a set of candidate concepts are identified. The objective of the disambiguation phase is then to select the concepts that most accurately highlight each mention's semantics, from among the concepts identified in the previous phase. There are generally four groups of work that perform disambiguation in annotators [1], namely popularity based, context based, collective disambiguation and graph-based techniques.

In the *popularity-based* approach the most frequently observed concept for a given mention is chosen. This method is usually combined with other approaches, since merely using this approach can lead to erroneous results. The reason is that the results do not consider context in which the mention appears and therefore largely ignore the main theme of the text. TagMe, Wikipedia Miner, AIDA, and Illinois Wikifier use the popularity-based approach combined with one of the following approaches for disambiguation.

Within the *context-based* approach, the context of the mention and the context of candidate concepts are compared. Context is typically modeled through bag-of-words and different distance measures [1]. Context-based approaches are used in DBpedia Spotlight, AIDA, and Illinois Wikifier for disambiguation.

The third type of disambiguation relies on *collective disambiguation*, where multiple mentions are disambiguated together. In this approach, target entities should be coherent and semantically related to each other. Many semantic annotation tools combine this approach with the popularity-based method such as TagMe, Wikipedia Miner, and Illinois Wikifier.

The final disambiguation approach is designed on a *graph-based* representation. In this approach, the extracted mentions and candidate concepts form the vertices of a graph. In this graph, the weighted edges between the mentions and candidate concepts represent the contextual similarity. On this basis, disambiguation is formulated as the task of finding a dense sub-graph in which each mention has exactly one edge. AIDA uses a graph-based approach for disambiguation.

In our work, disambiguation of a numeric value concerns the identification of the best matching property for that value from among the identified candidate properties in the detection phase. Our work is primarily based on the popularity-based approach. The selection of the best candidate is based on the cumulative distribution of values associated with each property in the knowledge base. The candidate property that has the closest distribution to the value observed in the given input text is selected.

2.3 Pruning

In this phase, the concepts that are irrelevant or marginally related to the topic of the input text are pruned. Some annotators such as AIDA perform this task in the disambiguation phase. However others such as DBpedia Spotlight perform it as a post-disambiguation phase.

In TagMe, pruning is based on the average value of each mention's link probability and the coherence between the selected concepts for all of the identified concepts. In DBpedia Spotlight pruning is based on a number of parameters that can be tuned by the user. Wikipedia Miner uses automated prunning similar to TagMe. It uses a topic detector to classify related and unrelated links in a document. Positive training instances for the classifier are the articles that were manually linked to an article in Wikipedia, while negative ones are those that were not. Features of these articles and the places where they were mentioned inform the classifier as to which mentions should or should not be linked. In our work, we do not perform pruning.

There are other areas of research that can be considered relevant to the theme of this paper including the work on ontology learning and knowledge base population. One of the state of the art automatic knowledge extraction tools is FRED [11]. This tool enables robust ontology learning and population (OL&P) from natural language. Ontology learning is the task of acquiring a domain model from a given text and therefore involves parsing of natural language and extracting complex relations and concepts for the purpose of taxonomy induction. FRED does the OL&P task based on Discourse Representation Theory (DRT).

3 Theoretical Model

The overall objective of our work is to find a best describing property and the concept that it belongs to for a quantitative value in a short text. We first describe the method for finding the best property and then explain how we identify its corresponding concept.

3.1 Property Identification

In order to find the most relevant property that accurately describes a numeric value[2], the first step is to identify the set of properties from the knowledge base that can potentially be related to that numeric value. Let us first provide a theoretic foundation for describing our work.

Definition 1 ***(Textual Snippet)*** *Let textual snippet $T = [w_1...w_k]$ be a string where w_i $(1 \leq i \leq k)$ is a word. We define $T.dt = w_j \in D$ and $T.r = w_{j-1}$ s.t. w_{j-1} is a numeric value and $2 \leq j \leq k$, and D is the set of all possible datatypes. Further we define, $T.S$ to be the set of all concepts that are spotted in T.*

According to this definition, our objective is to annotate $T.r$ with the most relevant property. For instance, for a textual snippet T such as "Motorola RAZR can support up to 64 MB", $T.dt$ is "MB" that represents the megabyte datatype and $T.r$ is "64". Furthermore, with the help of an automated semantic annotation system, one can find all the relevant concepts to T. For this example, $T.S$ is $\{Motorola\_Razr, Megabyte, Secure\_Digital\}$[3]. We rely on an existing annotator

[2] If numeric values are written in English words, we automatically convert them to numeric form before processing.

[3] In our work, we employ DBpedia as the source knowledge base; hence, the complete URI for the concepts would be in the form of `http://dbpedia.org/resource/Motorola_Razr`.

to provide the values for $T.S$. Now, our task is to find an appropriate property for the value “64” from the list of properties in our knowledge base (e.g. DBpedia).

Definition 2 ***(Knowledge Base)*** *Let $KB = \{c_1, ..., c_n\}$ be a knowledge base, where c_i $(1 \le i \le n)$ is a concept and $c_i.P = \{(p_1^{c_i}, v_1^{c_i}), ..., (p_m^{c_i}, v_m^{c_i})\}$ where $(p_j^{c_i}, v_j^{c_i})(1 \le j \le m)$ represents a property-value pair for concept c_i.*

For instance, for a concept such as “Motorola_A1000” in DBpedia, one can find a set of property-value pairs such as {(type, Device), (operatingsystem, “Symbian OS 7.0 + UIQ 2.1”), (storage, “24.0 megabyte”), ...}, among others. Based on Definitions 1 and 2, we formally specify the issue of property identification as follows:

Definition 3 ***(Property Identification)*** *For a knowledge base $KB = \{c_1, ..., c_n\}$ and a textual snippet T, let $P_c = \{p|(p, v) \in c.P\}$ be the set of all properties for concept c. The set of all possible properties in our knowledge base is defined as $UP = \bigcup_{c \in KB} P_c$. The objective is to find the most relevant property $p \in UP$ for $T.r$.*

In the context of the earlier example, our goal would be to find a relevant property for “64” which would in this case be “memory” or “storage”. As the first step we select a set of concepts from the knowledge base such that they consist of appropriate properties for $T.r$.

Definition 4 ***(Candidate Concepts)*** *For a textual snippet T, a Candidate concept set is defined as $C(T) = \{c|c \in KB, \exists(p, v) \in c.P \; s.t. \; v.dt = T.dt\}$ where $v.dt$ denotes the datatype for v.*

According to this definition, a candidate concepts set will include all concepts that have at least one property with a value whose datatype is equivalent to $T.dt$. In our running example, concept “Motorola_A1000” would be in the candidate concept set, since it has the datatype “megabyte” in the value of one of its properties. In order to choose the best concepts from the members of the Candidate concepts set a ranking function is required. We rank the members of the Candidate concepts set based on their distance to the spots in $T.S$.

Definition 5 ***(Concept Distance)*** *For concept c and textual snippet T, a distance function is defined as follows:*

$$dist(c, T) = \sqrt{\sum_{s \in T.S} \left(\frac{\rho(s)}{r(c, s) + \beta} \right)^2} \tag{1}$$

where semantic relatedness of two concepts c_1 and c_2 is represented as $r(c_1, c_2)$[4] *and ρ is the function that returns the confidence score of the mentioned concept in the text (provided by the annotator). Also β is a very small constant for when $r(c, s) = 0$.*

[4] We benefit from TagMe Relatedness API for this purpose in our experiments.

Table 1 shows a number of concepts and their distances to the spots in the context of the earlier example. We rank the concepts in the candidate concepts set using the distance function in Definition 5 and hypothesize that less distant concepts have a higher probability to include relevant properties for our purpose. Therefore, we select the top-k concepts from the candidate concept set, denoted by Top Concepts (TC)[5]. Based on the top-k concepts, the set of properties of concepts in TC that have a datatype equal to $T.dt$ will form a candidate property set defined as follows:

Table 1. Concept distances to the spots in $T.S$

Concept	**Distance**
Theatre_of_War_(video_game)	73.09
Sony_Ericsson_C510	61.63
Motorola_A1000	7.39

Definition 6 ***(Candidate Properties)*** *Assume TC is the set of top-k concepts based on the distance function in Equation 1. Candidate property set for TC is defined as* $CP(TC) = \{p|c \in TC, (p, v) \in c.P, v\ is\ Numeric\ and\ v.dt = T.dt\}$

In order to find the best related property from the candidate property set, for each of the properties in $CP(TC)$, a statistical analysis is done to see which property is more likely to have the numeric value $T.r$. To do so, we perform a statistical analysis over all observed values for each of the properties in CP(TC). In order to analyze the values of each property, we first build a set, called the Number Set.

Definition 7 ***(Number Set)*** *For a textual snippet T and a given property p, Number Set is defined as* $NS(p, T) = \{v_i|c \in KB, (p, v) \in c.P, r(v.dt, T.dt) > \alpha\}$.

The Number Set represents the set of all the numerical values for a specific property observed in the knowledge base as long as the datatype for that value had a semantic similarity score of above threshold α[6] with the datatype of the value that we are annotating ($T.dt$). Based on the Number Set, we calculate the relevance probability for a given property through its Cumulative Distribution Function (CDF). In CDF, we assume that a Number Set has a Gaussian distribution.

Definition 8 ***(CDF)*** *For a random variable R we have* $Pr[R \leq T.r] \approx CDF(T.r)$. *So,* $Pr[T.r - \Delta T.r < R < T.r + \Delta T.r] = CDF(T.r + \Delta T.r) - CDF(T.r - \Delta T.r)$ *where* $\Delta T.r = T.r/100$. *Therefore for a property p and a numeric value T.r,* $Pr(p, T.r) = CDF(NS(p, T), T.r + \Delta T.r) - CDF(NS(p, T), T.r - \Delta T.r)$.

[5] We set k to 10 in our experiments.

[6] In our experiments, we set α to 0.5.

The CDF for a property p and $T.r$ shows the probability of property p being the suitable representation for $T.r$. Table 2 shows a set of properties and their CDF values for the above example where the numeric value 64.0 was considered.

Table 2. The cumulative distance function for the properties'

Property	**CDF**
memory	0.004084854021963902
storage	1.0839821475783218E-4
size	1.316693881592279E-6

Based on the ranking provided through the CDF function, we are able to determine the best property that matches $T.r$. Algorithm 1 details the proposed approach to find the best property that describes a quantitative value mentioned in the input text. Lines 2-8 show how the candidate concepts set (C) is built. C is a subset of KB whose members (concepts) have a property value that includes the datatype of interest. After identifying candidate concepts, we find the top concepts (TC). TC is formed by taking the top-k members of C based on the ranking function in Definition 5 (line 9). Lines 10-16 show the process of forming the candidate properties set (CP). For every concept in the top concept set, all numeric-valued properties of the concepts that have a datatype close to $v.dt$ are chosen for CP. Finally, the property that has the highest probability of having $T.r$ as its value is identified as the property of interest (line 17).

Algorithm 1 IdentifyProperty(TexualSnippet T)

```
1: $C \leftarrow \emptyset$, $CP \leftarrow \emptyset$
2: for $c \in KB$ do
3:     for $(p, v) \in c.P$ do
4:         if $v$ is literal and $v.dt = T.dt$ then
5:             add $c$ to $C$
6:         end if
7:     end for
8: end for
9: TC $\leftarrow Top\_k(C, dist)$
10: for $c \in TC$ do
11:     for $(p, v) \in c.P$ do
12:         if $v$ is numeric and $r(v.dt, T.dt) > \alpha$ then
13:             add $p$ to $CP$
14:         end if
15:     end for
16: end for
17: return $\arg max_{p \in CP} Pr(p, T.r)$
```

3.2 Concept Identification

Now, given that the most relevant property for the numeric value is identified, in this phase, the objective is to find the most relevant concept mentioned in the text which either directly or through inference has the property identified in previous step. Let the identified property of the numeric value in our knowledge base be P. The objective is to find a subject for P with $T.r$ as its object. Note that the desired concept may or may not have the predicate (property) explicitly assigned to it. For example, Ford_XT_Falcon is a concept in the category of Ford_Falcon and it has the property "weight" in our knowledge base. However, Ford_Fairmont_(Australia) is in the category of Ford_Falcon but it does not have the property "weight". We are interested in all the concepts in $T.S$ that can potentially have P in one of their properties, which may not be direct but can be derived through hierarchical subclass inference.

Definition 9 ***(Candidate Mentions)*** *For a textual snippet T and a property P, a candidate mentions set is defined as $CM(T, P) = \{c|c \in T.S, (P, v) \in c.P\}$.*

In case candidate mentions set is empty (none of the mentioned concepts have the property P), we search for similar concepts in the knowledge base that have P as a property. A mention would be considered as a candidate, if there is at least a concept in the knowledge base that have the property P and is at least in one of the mention's categories as expressed in DBpedia's hierarchical concept categories. For example, Ford_XT_Falcon categories are Vehicles_introduced_in_1968, Cars_of_Australia, and Ford_Falcon. In order to identify the related concepts based on shared categories, we define the Related Concepts set as follows:

Definition 10 ***(Related Concepts)*** *For a concept s and a property P, related concepts set is defined as $RC(s, P) = \{c|c \in O, (P, v) \in c.P, cat(c) \cap cat(s) \neq \emptyset\}$ where cat is a function that returns the set of all DBpedia categories for a concept.*

Algorithm 2 shows the procedure for identifying the best concept for P. First, if the candidate mentions set is not empty, the concept in CM with the highest confidence (ρ) is selected (lines 1-3). Otherwise, we try to find other related concepts to each mention that have the property P. If such a concept is found, it will be added to CM (lines 5-9). CM is populated based on the related concepts. Finally, the best concept for property P is the one with the highest confidence value (ρ) in CM (line 10).

As an example in the earlier text "Motorola RAZR can support up to 64 MB" that was mentioned earlier, "memory" was selected as the best property for 64. Based on this identified property, there is only one concept in T.S = $\{Motorola\_Razr, Megabyte, secure\_Digital\}$, i.e. Motorola_Razr, that has "memory" as property. Therefore, Motorola_Razr would be the selected concept. In case there are more than one mentions that have the identified property, the one with the highest confidence is selected.

Algorithm 2 IdentifyConcept(TexualSnippet T, Property P)

```
1: if $CM(T, P)$ not empty then
2:     return $\arg max_{c \in CM(T,P)} \rho(c)$
3: end if
4: $CM \leftarrow \emptyset$
5: for $s \in T.S$ do
6:     if $RC(s, P)$ not empty then
7:         add $s$ to $CM$
8:     end if
9: end for
10: return $\arg max_{c \in CM} \rho(c)$
```

Now let us suppose that in the above example the “storage” was selected instead of “memory”. Then, in this case, candidate properties set would be empty, because none of the members of T.S has the “storage” property. Therefore, we need to consider the related concepts to the concepts in T.S. Here, there is only one concept, i.e., Motorola_Razr in T.S, which has a non-empty related concepts set. This is because we are able to find some concepts such as Motorola_Rokr that share a common DBpedia category with Motorola_Razr, i.e. Motorola_mobile_phones, and at the same time consist of the “storage” property. Therefore, our proposed algorithm identifies Motorola_Razr as the concept and “storage” as the property for the numeric value 64.

4 Experimental Results

In order to evaluate our work, we first developed a gold standard dataset that would include sentences that have quantitative values. Existing datasets that are used for evaluating semantic annotator systems were not suitable as they do not provide gold standard annotations for numeric values. Therefore, we recruited a group of ten Computer Science graduate students at MSc and PhD levels, all of whom had experience in working with semantic annotator systems before, to collect and annotate the gold standard dataset. The recruited graduate students were given a set of suggested concept-property pairs and were asked to collect descriptive sentences about each concept-property pair such that the sentences included quantitative content describing the desired property of the desired concept. Since our knowledge base (DBpedia) does not contain much numerical information about concepts, we provided the participants the suggested concept-property pairs to make sure that the collected gold standard would consist of concepts that exist in the knowledge base. Since the recruited graduate students were given a set of suggested concept-property pairs, there were no overlaps between the sentences they collected. The concept-property pairs were chosen so that they cover various domains including electronics, motor vehicles, movies & music, geographical locations, famous people and food. As a final step, all the collected gold standard content were processed by the TagMe semantic annotator and the extracted concepts were stored in the gold standard.

The developed gold standard dataset consists of 165 separate entries[7]. In the whole dataset, there are 1,225 unique concepts extracted by TagMe. In each instance, there are 9.85 mentioned concepts on average. Each of the entries was selected such that TagMe can find at least one spot for that entry.

With regards to DBpedia, in our experiments, we used DBpedia 3.8. locally installed on a MongoDB server and specifically exploited the "properties" collection. The "properties" collection has over 130 million subject-predicate-object triples. One of our observations when working with DBpedia was that although DBpedia is a great source of information, it does not provide substantial reliable numeric data. In other words, many of the properties that need to have numeric values are missing or have incorrect or too generic datatypes associated with them. Given DBpedia does not enforce a schema, we believe one of the areas that can be improved on this knowledge base is with regards to the quantitative values.

Based on the gold standard, our objective was to identify the correct concept and property for each of the quantitative values in the dataset entries. The experiments were run on a machine with 3.20 GHz CPU and 8 GB RAM. Table 3 (in Appendix) shows some sample entries and the corresponding concept and properties that were identified. In this table the mentioned entities are the spotted concepts extracted by TagMe. The predicted property and concept are those identified by our method for the highlighted numeric value in that dataset entry. As an example, in the first entry, fuelCapacity (`http://dbpedia.org/property/fuelCapacity`) is identified as the best property and Honda_Gyro (`http://dbpedia.org/resource/Honda_Gyro`) as the best concept for the numeric value 5.0L in the entry.

The experiments on the gold standard shows an accuracy of 73% for predicting the correct property and 72% for identifying the correct concept. It should be noted that given concept identification is dependent on the performance of the property detection method in our work, when the property was correctly identified, in 87% of the cases the concept was identified accurately as well.

One of the areas that we plan to investigate to further improve the performance of our work is to contextualize the consideration of properties with DBpedia categories. In other words, we intend to first identify the set of categories that a given input text belongs to, and then only consider property values of concepts within those categories for further predicting the correct property. For Example, if the input text is mainly about automobiles and a candidate property is "length", we would only consider values of "length" within concepts related to automobiles rather than "length" of irrelevant concepts such as rivers or cellphones.

[7] The dataset is publicly available at `http://ls3.rnet.ryerson.ca/people/mehrnaz/dataset.xlsx`.

5 Concluding Remarks

In this paper we have proposed a technique for semantically annotating quantitative values in textual content. To the best of our knowledge, our work is among the first to consider the semantic annotation of numerical values and connecting them to appopriate properties on an external knowledge base such as DBpedia. While we reach an overal accuracy of 73% on the gold standard, there is one main limitations for our work that we will be addressing in the future work: The core assumption of our work is that a numeric value is proceeded by a unit measure (datatype), e.g. 5.0 L. However, in many real world cases such a unit measure is non-existent after a numeric value. We are interested in predicting the unit measure of a numeric value based on its context.

References

1. Jovanovic, J., Bagheri, E., Cuzzola, J., Gasevic, D., Jeremic, Z., & Bashash, R. (2014). Automated Semantic Tagging of Textual Content, *IT Professional, 16*(6), 38-46.
2. Cornolti, M., Ferragina, P., & Ciaramita, M. (2013, May). A framework for benchmarking entity-annotation systems. In *Proceedings of the 22nd international conference on World Wide Web* (pp. 249-260). International World Wide Web Conferences Steering Committee.
3. Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011, September). DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems* (pp. 1-8). ACM.
4. Milne, D., & Witten, I. H. (2013). An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, 194, 222-239.
5. Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., ... & Weikum, G. (2011, July). Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (pp. 782-792). Association for Computational Linguistics.
6. Ratinov, L., Roth, D., Downey, D., & Anderson, M. (2011, June). Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (pp. 1375-1384). Association for Computational Linguistics.
7. Yi, M. (2008). Information organization and retrieval using a topic mapsbased ontology: results of a taskbased evaluation. *Journal of the American Society for Information Science and Technology*, 59(12), 1898-1911.
8. Wu, F., & Weld, D. S. (2010, July). Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 118-127). Association for Computational Linguistics.
9. Petasis, G., Karkaletsis, V., Paliouras, G., Krithara, A., & Zavitsanos, E. (2011, January). Ontology population and enrichment: State of the art. In *Knowledge-driven multimedia information extraction and ontology evolution* (pp. 134-166). Springer-Verlag.
10. Ferragina, P., & Scaiella, U. (2010, October). 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* (pp. 1625-1628). ACM.
11. Presutti, V., Draicchio, F., Gangemi, A. (2012). Knowledge extraction based on discourse representation theory and linguistic frames. In *Knowledge Engineering and Knowledge Management* (pp. 114-129). Springer Berlin Heidelberg.

Table 3. Some instances of dataset and their results

Dataset Entry	**Spotted Concepts by TagMe**	**Predicted Property**	**Predicted Concept**
The Honda Gyro is a family of small, three-wheeled, single-occupant vehicles sold primarily in Japan, and often used for delivery or express service with **5.0 L** fuel capacity.	Honda_Express, Japan, Fuel, Family, Engine_displacement, Tricycle, Single-occupant_vehicle, Honda_Gyro, Delivery_(commerce), Service_(economics)	fuelCapacity	Honda_Gyro
Before participating in the reality show, Sibuja Kaliraman weighed **90 kg**, and knew nothing about "presentation".	Reality_television, Kilogram, Sonika_Kaliraman, GMTV, Nothing	weight	Sonika_Kaliraman
Maple syrup is a syrup usually made from the xylem sap of sugar maple, red maple, or black maple trees, although it can also be made from other maple species. It contains **0.1 g** fat every 100 grams of the syrup.	Maple_syrup, Canadian_dollar, Xylem, Doepfer_A-100, Acer_nigrum, Gram, Fat, Acer_saccharum, Plant_sap, Species, Acer_rubrum, Maple	fat	Maple_syrup
The Boy Who Could Fly is a **114-minute** movie about an autistic boy who dreams of flying touches everyone he meets, including a new family who has moved in after their father dies.	Ontario_Highway_114, Dream, The_Boy_Who_Could_Fly, Autism, Minute, Film, Flight, Family, Death, Everyone_(film), Christian_Shephard, Father	runtime	The_Boy_Who_Could_Fly