Definition Extraction Using Sense-Based Embeddings

Luis Espinosa-Anke Horacio Saggion
TALN DTIC
Universitat Pompeu Fabra
Carrer Tànger 122-134
08018 Barcelona, Spain
{luis.espinosa,horacio.saggion}@upf.edu

Claudio Delli Bovi
Dipartimento di Informatica
Sapienza Università di Roma
Viale Regina Elena 295
00161 Roma, Italy
dellibovi@di.uniroma1.it

Abstract: Definition Extraction is the task to identify snippets of free text in which a term is defined. While lexicographic studies have proposed different definition typologies and categories, most NLP tasks aimed at revealing word or concept meanings have traditionally dealt with lexicographic (encyclopedic) definitions, for example, as a prior step to ontology learning or automatic glossary construction. In this paper we describe and evaluate a system for Definition Extraction trained with features derived from two sources: Entity Linking as provided by Babelfy, and semantic similarity scores derived from sense-based embeddings. We show that these features have a positive impact in this task, and report state-of-the-art results over a manually validated benchmarking dataset.

Keywords: Embeddings, Entity Linking, Definition Extraction, Information Extraction

1 Introduction

Definitions are fundamental sources for retrieving the meaning of terms (Navigli and Velardi, 2010). However, looking them up manually in naturally occurring text is unfeasible. For this reason, automatic extraction of definitional text snippets is on demand, especially for tasks like Ontology Learning (Velardi, Faralli, and Navigli, 2013; Snow, Jurafsky, and Ng, 2004; Navigli and Velardi, 2006), Question Answering (Saggion and Gaizauskas, 2004; Cui, Kan, and Chua, 2005), Glossary Creation (Muresan and Klavans, 2002; Park, Byrd, and Boguraev, 2002), or support for eLearning environments (Westerhout and Monachesi, 2007).

The task to automatically identify definitions in free text is Definition Extraction (DE). As in many extraction tasks in NLP, a great deal of previous work has relied on linguistic patterns. For instance, by directly identifying verbal cue phrases (Rebeyrolle and Tanguy, 2000; Saggion and Gaizauskas, 2004; Sarmento et al., 2006; Storrer and Wellinghoff, 2006). Moreover, machine learning approaches have incorporated linguistic patterns as information for training classifiers. For instance, (Navigli and Velardi, 2010) propose a generalization of word lattices for the tasks of DE and Hypernym Extraction. In addition, (Boella et al., 2014) exploit syntactic dependencies to create word representations, which are used as features for training an SVM classifier. Moreover, (Jin et al., 2013) use hand-crafted shallow parsing patterns in a CRF-based sequential labeller for DE in scientific papers. Finally, (Espinosa-Anke and Saggion, 2014) take advantage of syntactic dependencies in the form of a bag-of-subtrees approach together with metrics exploiting the dependency tree such as a word's degree or the part-of-speech of its children.

Although the systems reported above achieve competitive results, in none of them semantic information is used, opening therefore clear avenues for improvement. We hypothesize that external knowledge can contribute dramatically to the DE task, and can be also useful for potential cross-domain or multilingual experiments. In this paper, rather than introducing knowledge from structured resources, we leverage SensEmbed (Iacobacci, Pilehvar, and Navigli, 2015), a recent work that applies state-of-the-art representation techniques for modelling individual word senses. Our choice stems from the intuition that sense-based representations can reveal properties of *semantic compactness*, which may be indicators of definitional or gloss-like text snippets.

In the next section we proceed to describe our approach to DE.

2 DE Using SensEmbeddings

2.1 Data

We perform our experiments on the WCL dataset (Navigli, Velardi, and Ruiz-Martínez, 2010), a subset of Wikipedia containing 1717 definitions (coming from the first sentence of randomly sampled Wikipedia articles), and 2847 of what the authors called "syntactically plausible false definitions", i.e. sentences with a syntactic structure similar to that of a definition, and where the defined term appears explicitly, but are not definitions.

2.2 Entity Linking

The first step of our approach consists in running Babelfy (Moro, Raganato, and Navigli, 2014), a state-of-the-art WSD and Entity Linking tool, over the WCL dataset. In this way, we obtain disambiguations for content text snippets, which are used to build a semantically rich representation of each sentence. Consider the following definition and its concepts, represented with their corresponding BabelNet (Navigli and Ponzetto, 2012) synset id:

The⟨O⟩ Abwehr⟨01158579n⟩ was⟨O⟩ a⟨O⟩ German⟨00103560a⟩ intelligence⟨00047026n⟩ organization⟨00047026n⟩ from⟨O⟩ 1921⟨O⟩ to⟨O⟩ 1944⟨O⟩.

This disambiguation procedure yields two important pieces of information. On one hand, the set of concepts, represented as BabelNet synsets, e.g. the synset with id bn:01158579n for the concept Abwehr$_{bn}$[1]. On the other hand, we also obtain a set of non-disambiguated snippets (either single word or multiword terms), which can be also used as indicators for spotting a definitional text fragment in a corpus (from the above example: {*the*, *was a*, *from 1921 to 1944*}).

2.3 Sense-Based Embeddings

SensEmbed works in two main steps: First, a large text corpus is disambiguated with Babelfy. Then, *word2vec* (Mikolov, Yih, and Zweig, 2013; Mikolov et al., 2013) is applied to the disambiguated corpus, yielding a vectorial latent representation of word senses. This enables a disambiguated vector representation of concepts. For instance, for the term "New York" (BabelNet id bn:00041611n), there are vectors for lexicalizations such as "NY", "New York", "Big Apple" or even "Fun City".

We use SensEmbed for computing the semantic similarity among concepts in each sentence of the WCL corpus. These similarities are afterwards used for computing features that will serve as input for a sentence-based classifier. We denote in the rest of this paper the semantic similarity between two concepts x and y as $\textsc{sim}(x, y)$, which is simply the cosine similarity of the closest vectors associated to their corresponding lexicalizations. Formally, let L be the set of lexicalizations included in SensEmbed and Γ the set of associated vectors to each lexicalization. We compute sim as follows: (1) Retrieve all the available lexicalizations in L of both x and y, namely $L(x) = \{s_x^1, ..., s_x^m\}$ and $L(y) = \{s_y^1, ..., s_y^z\}$. (2) Next, retrieve from Γ the corresponding sets of vectors $V(x) = \{v_x^1, ..., v_x^m\}$ and $V(y) = \{v_y^1, ..., v_y^z\}$. (3) Finally, we compare each possible pair of senses and select the one maximizing the cosine similarity cos between the corresponding vectors, i.e.

$$\textsc{cos}\,(x, y) = \max_{v_x \in V(x), v_y \in V(y)} \frac{v_x \cdot v_y}{||v_x||||v_y||}$$

For example, given the definition of the term *bat*, "A bat is a mammal in the order Chiroptera", we obtain a set D of three concepts: bat$_{bn}$, mammal$_{bn}$ and Chiroptera$_{bn}$. For each pair of concepts $c_1, c_2 \in D$, we compute $\textsc{sim}(c_1, c_2)$, and perform this operation over all pairs in D.

Table 1 shows the sim representation of this definition (d) and one non-definitional sentence (n) also referring to *bat*: "This role explains environmental concerns when a bat is introduced in a new setting". Note the higher sim scores for concept pairs in the definitional sentence (in bold). Also, note that since the non-definition is less *semantically compact*, our procedure assigned to the term *bat* vectors corresponding to the programming language *batch*, or to *batch* files.

[1] For clarity, we use the subscript *bn* to refer to the concept's BabelNet id, rather than using the actual numeric id.

Vector	Vector'	SIM
bat$_d$	**mammal**$_d$	0.59
bat$_d$	**chiroptera**$_d$	0.29
mammal$_d$	**chiroptera**$_d$	0.31
role$_n$	environmental_concern$_n$	0.21
purpose$_n$	batch_language$_n$	0.15
environmental_concern$_n$	role$_n$	0.21
conservation_group$_n$	batch_file$_n$	0.12
batch_language$_n$	purpose$_n$	0.15
batch_file$_n$	conservation_group$_n$	0.12

Table 1: Representation of a definition and a non-definition in terms of the similarities of its concepts.

In the remainder of the paper, the whole set of similarity scores over a given sentence, obtained with this strategy, is denoted as Δ.

2.4 Features

We design three types of features: (1) Bag-of-Concepts; (2) Bag-of-non-disambiguated text snippets; and (3) Similarity metrics over Δ. These features are then used to train different classification algorithms, whose performance is evaluated in 10-fold cross validation.

Bag-of-Concepts

We extract the 100 most frequent BabelNet synsets in the training data, and generate a feature vector for each one. Each feature has a binary value, either *True* or *False*, referring to whether such synset was found in the sentence to be classified. In most folds, the most frequent synsets refer to ancient languages such as Greek or Latin, or to scientific disciplines such as Maths or Computer Science. This reveals that presence of these concepts in a sentence is a strong indicator of such sentence of being a definition in the encyclopedic genre.

Bag-of-non-Disambiguated Concepts

We extract the 500 most frequent text snippets that Babelfy did not disambiguate. The vector construction procedure is the same as in Bag-of-Concepts. In this case, we obtain results consistent with previous studies in that the pattern "is a" is the most frequent and hence a feature with high predictive power, followed by "is the", "of a" and "is any".

Semantic Features

We put forward a novel set of features stemming from the hypothesis that, in a definition, most concepts should be closely related, and hence should show higher semantic similarity than *distractor* sentences. For instance, in our working example "A bat is a mammal in the order Chiroptera", the concepts *bat*, *mammal* and *Chiroptera* are closely related, and intuitively their corresponding vectors should be *more compact* and closer in the vector space, as opposed to one of its distractors in the WCL corpus: "This role explains environmental concerns when a bat is introduced in a new setting". Here, concepts like *bat*, *to explain*, *environmental* or *setting* have a set of associated vectors *more sparsely distributed* in the vector space.

We build on this intuition to propose the following features:

- **AllSims** The sum of the SIM scores in Δ.
- **AvgSims** The average of the SIM scores in Δ.
- **AvgBiggestSubGraph** We can express our list of SIM scores as a non-directed cyclic graph, in which each node is a concept and each edge is weighted according to their SIM score. However, there are cases in which not all components of the graph are connected because one concept may be associated to two different lexicalizations depending on which concept it is disambiguated against. For instance, the concept for *mammal* in our working example may be lexicalized as *mammal* if disambiguated against *bat*, and as *mammalia* if disambiguated against *chiroptera.* This feature is the average of the cosine scores of the biggest connected subgraph generated from Δ[2]. Note that if the sentence graph is complete, **AvgSims** and **AvgBiggestSubGraph** yield the same score.
- **TopDegreeScore** First, we obtain the node with highest degree in the graph representation described above, i.e. the most connected node. Then, we compute the average SIM score over this node and its neighbours. We hypothesize that this measure should reward concepts whose disambiguation remains the same regardless of the concept they are disam-

[2] Graph operations performed in our experiments were done with the Python library NetworkX: https://networkx.github.io/

biguated against, which can be seen as another *semantic compactness* measure.

- **NumEdges** The number of edges of the graph described above. As the disambiguation options for a given concept increases, so will increase the number of edges of the graph representation. This is a feature aimed at capturing *non-definitional* sentences.
- **MaxScore and MinScore** The maximum and minimum sim score among all the concept pairs in Δ. We hypothesize that in a definitional sentence, there will be at least one pair highly similar, the one between the defined term and the hypernym.

These features are used to perform a set of experiments with the machine learning toolkit Weka (Witten and Frank, 2005). While many configurations and algorithms were tested, for brevity we report here the ones for the best performing experiment, based on Support Vector Machines.

3 Evaluation

Our approach (Our) shows competitive results, outperforming previous systems on the same dataset. We compare against three main competitors: (1) The WCL algorithm (WCL), which generalizes word-lattices over surface form and part-of-speech tags, hence producing word-class lattices (Navigli and Velardi, 2010); (2) A supervised machine-learning setting (BdC) in which syntactic dependencies are used to construct word representations in terms of their direct descendants (Boella et al., 2014); and (3) Another supervised approach (EspSag) also based on syntactic dependencies, but representing each sentence as a bag-of-dependency-subtrees (Espinosa-Anke and Saggion, 2014).

As is the case in all the systems described, performance is evaluated with the classic Precision, Recall and F-Score measures at sentence-level. Table 2 shows the performance of all systems.

We complement our experiments by evaluating the relevance of each individual feature from our feature set. To this end, we compute their Information Gain score, which measures the decrease in entropy when the feature is given vs. absent (Forman, 2003). The feature ranking provided in Table 3 shows the discriminative power of the features derived from SensEmbed, reinforcing our claim that semantic information can be effectively applied to the DE task.

	Precision	**Recall**	**F-Score**
WCL	**98.8**	60.7	75.2
BdC	88.1	76.2	81.6
EspSag	85.9	85.3	85.4
Our	86.1	**86.0**	**86.0**

Table 2: Comparative results over the WCL dataset.

InfGain Score	**Feature**
"Contains:is_a"	0.19
AvgSims	0.13
AvgBiggestSubGraph	0.12
MaxScore	0.07
MinScore	0.06
TopDegreeScore	0.04
"Contains:is_an"	0.03
"Contains:bn00103785a"	0.02
NumEdges	0.01
AllSims	0.01

Table 3: Top 10 features according to their Information Gain score

4 Conclusions

Identifying definitional text snippets in free text is a task that can be integrated in more complex systems on ontology learning, dictionary or glossary construction, or for supporting terminological or eLearning applications. In this paper, we have described a supervised approach to DE that benefits substantially from introducing simple metrics derived from SensEmbed, a sense-based vector representation of concepts and their lexicalizations. For future work, we would like to introduce features derived from the BabelNet graph, such as proximity, random walks or relation type; as well as adding additional vector comparison measures, e.g. the Tanimoto coefficient, used in (Iacobacci, Pilehvar, and Navigli, 2015).

References

Boella, Guido, Luigi Di Caro, Alice Ruggeri, and Livio Robaldo. 2014. Learning from syntax generalizations for automatic semantic annotation. *Journal of Intelligent Information Systems*, pages 1–16.

Cui, Hang, Min-Yen Kan, and Tat-Seng Chua. 2005. Generic soft pattern models for definitional question answering. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 384–391. ACM.

Espinosa-Anke, Luis and Horacio Saggion. 2014. Applying dependency relations to definition extraction. In *Natural Language Processing and Information Systems*. Springer, pages 63–74.

Forman, George. 2003. An extensive empirical study of feature selection metrics for text classification. *The Journal of machine learning research*, 3:1289–1305.

Iacobacci, Ignacio, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. Sensembed: Enhancing word embeddings for semantic similarity and relatedness. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Beijing, China, July. Association for Computational Linguistics.

Jin, Yiping, Min-Yen Kan, Jun-Ping Ng, and Xiangnan He. 2013. Mining scientific terms and their definitions: A study of the ACL anthology. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 780–790, Seattle, Washington, USA, October. Association for Computational Linguistics.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751. Citeseer.

Moro, Andrea, Alessandro Raganato, and Roberto Navigli. 2014. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244.

Muresan, A and Judith Klavans. 2002. A method for automatically building and evaluating dictionary resources. In *Proceedings of the Language Resources and Evaluation Conference (LREC*.

Navigli, Roberto and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Navigli, Roberto and Paola Velardi. 2006. Ontology enrichment through automatic semantic annotation of on-line glossaries. In *Managing Knowledge in a World of Networks*. Springer, pages 126–140.

Navigli, Roberto and Paola Velardi. 2010. Learning word-class lattices for definition and hypernym extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1318–1327, Stroudsburg, PA, USA. Association for Computational Linguistics.

Navigli, Roberto, Paola Velardi, and Juana María Ruiz-Martínez. 2010. An annotated dataset for extracting definitions and hypernyms from the web. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).

Park, Youngja, Roy J. Byrd, and Branimir K. Boguraev. 2002. Automatic Glossary Extraction: Beyond Terminology Identification. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Morristown, NJ, USA. Association for Computational Linguistics.

Rebeyrolle, Josette and Ludovic Tanguy. 2000. Repérage automatique de structures linguistiques en corpus : le cas des énoncés définitoires. *Cahiers de Grammaire*, 25:153–174.

Saggion, Horacio and Robert Gaizauskas. 2004. Mining on-line sources for definition knowledge. In *17th FLAIRS*, Miami Bearch, Florida.

Sarmento, Luís, Belinda Maia, Diana Santos, Ana Pinto, and Luís Cabral. 2006. Corpógrafo V3 From Terminological Aid to Semi-automatic Knowledge Engineering. In *5th International Conference on Language Resources and Evaluation (LREC'06)*, Geneva.

Snow, Rion, Daniel Jurafsky, and Andrew Y Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. *Advances in Neural Information Processing Systems 17*.

Storrer, Angelika and Sandra Wellinghoff. 2006. Automated detection and annotation of term definitions in German text corpora. In *Conference on Language Resources and Evaluation (LREC)*.

Velardi, Paola, Stefano Faralli, and Roberto Navigli. 2013. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707.

Westerhout, Eline and Paola Monachesi. 2007. Extraction of Dutch definitory contexts for elearning purposes. *Proceedings of the Computational Linguistics in the Netherlands (CLIN 2007), Nijmegen, Netherlands*, pages 219–34.

Witten, Ian H and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.