# Weasel: a machine learning based approach to entity linking combining different features

Felix Tristram, Sebastian Walter, Philipp Cimiano, and Christina Unger

Semantic Computing Group, CITEC, Bielefeld University

**Abstract.** The task of entity linking consists in disambiguating named entities occurring in textual data by linking them to an identifier in a knowledge base that represents the real-world entity they denote. We present *Weasel*, a novel approach that is based on a combination of different features that is trained using a Support Vector Machine. We compare our approach to state-of-the-art tools such as FOX and DBpedia spotlight, showing that it outperforms both on the AIDA/CoNLL dataset and provides comparable results on the KORE50 dataset.

**Keywords:** Wikipedia, entity linking, vector similarity

## 1  Introduction

The task of entity linking consists in disambiguating named entities occurring in textual data by linking them to an identifier in a knowledge base that represents the real-world entity they denote. We present *Weasel*, a novel approach that is based on a combination of different features that is trained using a Support Vector Machine. Given a named entity and a resource candidate, Weasel combines four features that quantify i) the likelihood that the actual name refers to the resource candidate (extracted from Wikipedia anchors), ii) the similarity of the textual context in which the name occurs to the textual context of the resource candidate (measured as the bag-of-words similarity using cosine), iii) the overlap between the semantic neighborhood signature of the resource candidate and the named entities mentioned in the larger context of the named entity, and iv) the prominence of the resource candidate as measured by its page rank in the DBpedia graph. An optimal model is trained using an SVM which combines these four features into an overall decision whether the named entity refers to a candidate resource or not.

As underlying knowledge base, we use DBpedia and rely on Wikipedia pages as textual context of the resources in DBpedia to extract our textual features.

We compare our approach to state-of-the-art tools such as FOX [6] and DBpedia spotlight [3], showing that it outperforms both on the AIDA/CoNLL dataset and provides comparable results on the KORE50 dataset.

In the following section we describe the approach and then report evaluation results in Section 3.

## 2   Approach

The input data for Weasel is a regular piece of text, for example a newspaper article in which entity-representing sequences of tokens (*fragments*) are already identified (we use the Stanford NER [2]). For a given fragment, the algorithm's task is to determine the matching real-world entity. It does so by picking the most likely entity from a set of entity candidates, e.g. specified in a knowledge base, in our case DBpedia.

A fragment which represents a real-world entity is called an *anchor*. To gather a database of anchors and their corresponding entities, we extracted all anchor texts from a Wikipedia dump[1]. Within a dump, hyperlinks to other pages appear in the form `[[Barack Obama|President Obama]]`, where `Barack Obama` refers to an entity[2] and `President Obama` is the anchor string which appears in the page's text. A map is generated by parsing all Wikipedia articles (in a chosen language) and counting the occurrences of anchor-entity pairs. The candidates for a given fragment are found by looking up all entities that were referred to by the fragment within Wikipedia.

Candidates are scored by a support vector machine.[3] Four features are computed for each candidate, which are used as input for the SVM: tf-idf vector similarity, semantic neighborhood signature vector similarity, candidate reference probability, and page rank. In the following we explain each of them in more detail.

### 2.1   tf-idf vector similarity

The **tf-idf vector similarity** measures how similar the input is to the abstract of the candidate's Wikipedia entry with regards to the words which appear in both texts. The tf-idf value for a word reflects how important it is in a document, therefore if there is a large overlap between the important words in the input and the abstract of a candidate's Wikipedia page, the candidate is considered to be more likely correct. It is calculated based on the term frequency $\text{tf}(t, d)$, defined as the raw frequency of term $t$ in document $d$, and the inverse document frequency $\text{idf}(t, D)$, which is defined as follows:

$$\text{idf}(t, D) = \log \frac{N}{|d \in D : t \in d|}$$

Where $N$ is the number of documents in the corpus and $|d \in D : t \in d|$ is the number of documents $t$ appears in. The tf-idf value is the product of the two:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

---

[1] `http://en.wikipedia.org/wiki/Wikipedia:Database_download`

[2] `http://en.wikipedia.org/wiki/Barack_Obama`

[3] For the training set the correct assignment and three randomly chosen false candidates for each entity are used. Weasel is using Weka's SMO: `http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/SMO.html`

Weasel computes a tf-idf vector for each entity extracted from the Wikipedia dump by picking those 100 terms $t$ from the abstract $a$ of its Wikipedia page that have the highest tf-idf score. Here the tf-value can be obtained by simply counting all occurrences of $t$ in $a$, and the idf-value can be obtained by treating the set of all entity abstracts as the corpus and computing the idf-score for every occurring word.

The tf-idf vector for the input is then computed in the same fashion, where the term frequency for every term in the input is determined by counting its occurrences in the input text. Terms that are unique in the input are ignored, as they do not affect the entity's vector score (they are multiplied by zero in the vector similarity calculation).

The tf-idf vector similarity is then calculated as the cosine similarity between two vectors $A$ (the input terms) and $B$ (the candidate terms):

$$\text{tfidfVectorSimilarity}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

## 2.2   Semantic Neighborhood Signature Vector Similarity

The *semantic neighborhood signature vector similarity* measures the overlap between the candidate's *semantic neighborhood signature* and the set of all candidates for the input. The rationale is that a candidate is more likely to be the correct assignment for a given fragment if entities from its semantic neighborhood signature are considered for other fragments (as related entities probably appear together).

The semantic neighborhood signature for an entity is computed in a similar way as Babelfy [4] does, except that instead of the BabelNet[4] knowledge base we use a graph of interlinked Wikipedia pages, where each page is a vertex and each link between pages is an edge. The idea is to identify for each resource all other resources it is closely related to. This is a two step process. First, the edges are assigned weights so that edges in denser parts of the graph become more important. The weight of an edge $\text{weight}(v, v')$ is the number of directed triangles (directed cycles of length 3) it appears in:

$$\text{weight}(v, v') := |\{(v, v', v'') : (v, v'), (v', v''), (v'', v) \in E\}| + 1$$

Where $v$, $v'$ and $v''$ are vertices and $E$ is the set of all the graph's edges.

Second, random walks with restart are performed from each vertex. For a number of $n$ steps (in our case $100\,000$, which are still computable within a reasonable timespan) the graph is explored by 'walking' to a connected vertex with probability $1 - \alpha$ or to move back to the vertex of origin with probability $\alpha$. The probability of visiting the vertex $v'$ is based on the normalized weight of the connecting edge:

$$P(v'|v) = \frac{\text{weight}(v, v')}{\sum\limits_{v'' \in V} \text{weight}(v, v'')}$$

---

[4] http://babelnet.org/

Where V is the set of all vertices in the graph.

During the random walk for a given root vertex $v_r$, the visits to each vertex $v_i (i \neq r)$ are counted. The result is a frequency distribution of related vertices in which the vertices with higher counts are interpreted as to be more relevant to the root vertex than vertices with lower counts. Vertices with a count lower than threshold $\tau$ are ignored[5].

We compute a measure similar to the *personalized page rank* [1] using a random walk approach similar to the one implemented in Babelfy. This step is performed offline.

To calculate the similarity score for each candidate, their semantic neighborhood signature vector needs to compared to the semantic neighborhood signature vector of the input. Unfortunately, the input does not have a semantic neighborhood signature of its own. As a surrogate, a candidate vector is generated as follows: Let $m(e) \rightarrow c$ be a mapping from an entity $e$ to a count $c$. For each candidate entity of every fragment, check whether it already appears in the mapping. If not, add it to the map and set its count to one. If it exists, increment its count by one.

Therefore a candidate's semantic neighborhood signature similarity score increases as entities from their semantic neighborhood signature appear more often as candidates in a given input.

### 2.3   Candidate Reference Probability

To account for how often an anchor refers to a given entity, the *candidate reference probability* is incorporated into the score calculation. For example, for the anchor *Michelangelo* we might receive as candidates both the artist Michelangelo[6] and the Ninja Turtle Michelangelo[7]. While both are valid candidates worthy of consideration, we can extract from Wikipedia that *Michelangelo* referred only 21 times to the Ninja Turtle, but 1173 times to the artist. Turning these candidate reference frequencies into probabilities (dividing by the total number of references towards a given entity) we can use them as a factor when calculating the score for a candidate:

$$\mathrm{crp}(a, e) := \frac{\mathrm{ref}(a, e)}{\sum\limits_{a' \in A} \mathrm{ref}(a', e)}$$

Where $\mathrm{ref}(a, e)$ is the number of references from anchor/candidate $a$ to entity $e$ and $A$ is the set of all anchors referring to $e$.

---

[5] $\tau = 10$ in our implementation, so that $n$ and $\tau$ have the same ratio as in Babelfy.

[6] http://en.wikipedia.org/wiki/Michelangelo

[7] http://en.wikipedia.org/wiki/Michelangelo_(Teenage_Mutant_Ninja_Turtles)

**2.4   Page Rank**

The idea behind the PageRank algorithm is to estimate the relevance of a website to a search query not only by whether the search terms appear on that page, but also by how many other pages were providing a hyperlink to the website in question. As DBpedia entries form a highly interconnected graph, the same principle can be applied in order to determine the page rank of a resource in DBpedia.

Let $u$ be a web page. $B_u$ is the set of pages that have a hyperlink pointing to $u$. Let $N_u$ be the number of outgoing links for page $u$. A slightly simplified PageRank is then defined as follows:

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v}$$

Where $c$ is a normalizing factor. Notably the PageRank of a page is derived from the PageRank of its incoming links, meaning that the algorithm has to be iterated until the system reaches a steady state. The overall PageRank for a graph is supposed to be constant, but there are corner cases (like cycles) that have to be accounted for; see for more information. Our implementation uses the JUNG framework[8].

## 3   Results

We evaluate our entity linking approach on two datasets[9], AIDA/CoNLL and KORE50, and compare its results with two other state-of-the-art entity linking approaches, AGDISTIS [7] as part of FOX[10], and DBpedia Spotlight[11], in terms of precision and recall. Precision, recall, and the derived $F_1$ score were defined as follows:

$$Precision = \frac{\text{\# of correctly assigned entities}}{\text{\# of all assigned entities}}$$

$$Recall = \frac{\text{\# of correctly assigned entities}}{\text{\# of gold standard entities}}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Precision, Recall and F-Measure are micro-averaged over all named entities. Documents in which more named entities occur have thus a larger impact on the scores.

---

[8] Java Universal Network/Graph Framework: `http://jung.sourceforge.net/`

[9] Weasel's results for the 10-fold cross validations can be downloaded from: `http://sebastianwalter.org/weasel/nFoldResultsAIDA.zip` and `http://sebastianwalter.org/weasel/nFoldResultsKore50.zip`

[10] `http://aksw.org/Projects/FOX.html`

[11] `http://spotlight.dbpedia.org`

|  | AIDA/CoNLL | | | KORE50 | | |
|---|---|---|---|---|---|---|
|  | Precision | Recall | $F_1$ | Precision | Recall | $F_1$ |
| FOX | 0.49 | 0.30 | 0.37 | **0.30** | 0.21 | 0.25 |
| DBpedia Spotlight | 0.22 | 0.46 | 0.30 | 0.22 | **0.34** | 0.27 |
| Weasel | **0.60** | **0.57** | **0.58** | **0.30** | 0.26 | **0.28** |

Table 1: Results for the AIDA/CoNLL and KORE50 datasets.

The quality of the NER used to find entities poses an upper bound on the achievable entity linking scores. For the AIDA/CoNLL dataset the highest achievable linking precision is 0.79, while the highest achievable recall is 0.98 (max. $F_1$: 0.88). For the KORE50 dataset the values are 0.93 and 0.88 respectively (max. $F_1$: 0.90).

Table 1 shows the evaluation results. Weasel outperforms both FOX and DBpedia Spotlight on the AIDA/CoNLL dataset and achieves comparable results on the KORE50 dataset. The latter is due to the fact that KORE50 was designed to have a high degree of ambiguity and short sentences, which renders Weasel's approach less effective: Short sentences provide fewer tokens to evaluate a tf-idf overlap, while increased numbers of candidates allow for more detrimental semantic neighborhood signature vector overlaps.

Table 2 shows results leaving out single features and using only one feature. We clearly see that the page rank is the dominating factor. This deserves closer investigation.

Weasel works best when a fragment refers to the most common meaning of a word. For example, assignments like $BSE \rightarrow$ `Bovine_spongiform_encephalopathy` are usually correct. More ambiguous fragments get assigned correctly as well when that assignment is the most common one in the given context, for example *commission* $\rightarrow$ `European_Commission` in a debate of the European Parliament. Problems arise when the target entity is not the most common assignment for a given fragment. For example, the fragment *Spanish* in *Spanish Farm Minister Loyola de Palacio* leads to the assignment *Spanish* $\rightarrow$ `Spanish_Language`, as the fragment usually refers to the language, whereas the gold assignment is *Spanish* $\rightarrow$ `Spain`.

## 4    Conclusion

We have presented Weasel, a new entity linking approach that exploits a combination of features that are combined using a Support Vector Machine. We have shown that the approach works very well when the input sentences are longer, clearly outperforming FOX and DBpedia Spotlight on the AIDA/CoNLL dataset. On the KORE50 dataset where sentences are shorter, it achieves comparable results to these approaches. In future work, we will release a service that can be publicly accessed using a similar interface to DBpedia Spotlight and

| | AIDA/CoNLL | | | KORE50 | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | $F_1$ | Precision | Recall | $F_1$ |
| *without* | | | | | | |
| SNS | 0.60 | 0.57 | 0.59 | 0.32 | 0.28 | 0.29 |
| tf-idf | 0.60 | 0.57 | 0.58 | 0.32 | 0.28 | 0.29 |
| page rank | 0.36 | 0.34 | 0.35 | 0.12 | 0.10 | 0.11 |
| CRP | 0.62 | 0.60 | 0.61 | 0.31 | 0.27 | 0.29 |
| *only with* | | | | | | |
| SNS | 0.36 | 0.35 | 0.36 | 0.07 | 0.06 | 0.06 |
| tf-idf | 0.22 | 0.21 | 0.22 | 0.08 | 0.07 | 0.07 |
| page rank | 0.62 | 0.60 | 0.61 | 0.32 | 0.28 | 0.29 |
| CRP | 0.21 | 0.20 | 0.21 | 0.08 | 0.07 | 0.07 |

Table 2: Results for the AIDA/CoNLL and KORE50 datasets when one of the components is removed and when only one component is used for classification. SNS is Semantic Neighborhood Signature Vector Similarity, CRP is Candidate Reference Probability.

FOX. We will also extend the approach to other languages and perform a more systematic evaluation using the GERBIL framework [8].

## Acknowledgment

## References

1. Eneko Agirre and Aitor Soroa. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 33–41, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
2. Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.
3. Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.
4. Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the Association for Computational Linguistics (TACL)*, 2:231–244, 2014.

5. Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: bringing order to the web. 1999.
6. René Speck and Axel-Cyrille Ngonga Ngomo. Named entity recognition using fox. In *International Semantic Web Conference 2014 (ISWC2014), Demos & Posters*, 2014.
7. Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Michael Rder, Daniel Gerber, SandroAthaide Coelho, Sren Auer, and Andreas Both. Agdistis - graph-based disambiguation of named entities using linked data. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandei, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web - ISWC 2014*, volume 8796 of *Lecture Notes in Computer Science*, pages 457–471. Springer International Publishing, 2014.
8. Ricardo Usbeck, Michael Röder, Axel-Cyrille Ngonga Ngomo, Ciro Baron, Andreas Both, Martin Brümmer, Diego Ceccarelli, Marco Cornolti, Didier Cherix, Bernd Eickmann, et al. Gerbil: General entity annotator benchmarking framework. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1133–1143. International World Wide Web Conferences Steering Committee, 2015.