Institut für Computerlinguistik

March 27, 2018, Heidelberg

Course
Formal Semantics
Lecturers
Dr. Vivi Nastase
Dr. Michael Herweg

Author
Lyubomira Dimitrova
dimitrova@cl.uni-heidelberg.de
Matriculation number
3443834

Web Search Result Clustering

1 Introduction

This project is based on the SemEval-2013 Task 11 (*Evaluating Word Sense Induction and Disambiguation within An End-User Application* [1]). The task is to develop a system which clusters the top 100 search engine results to an ambiguous query, according to the query's sense. A clustering can be achieved implicitly by a word sense disambiguation (WSD) classification approach based on a sense inventory, or explicitly, by applying a word sense induction (WSI) algorithm to raw text.

I propose a simple WSI approach to this web search result clustering task, making use exclusively of the search result snippets.

2 Idea & Implementation

The WSI approach to the SemEval-2013 Task 11 can be reduced to two distinct problems. First, obtaining sensible representations of the search result snippets, and secondly, clustering them with respect to a relevant similarity measure (depending on the representations and the chosen clustering algorithm).

2.1 Preprocessing

First, I extracted the queries from the topic.txt file, using the following representation:

$$45 \text{ soul\_food} \rightarrow 45 : [soul, food, soul\_food]$$

Then, I used the spaCy Python library to process the search results file (results.txt), extracting for each snippet a bag of words, heavily filtered to reduce noise. Each BoW consists of all lemmatized snippet nouns longer than three characters, which are not the query or part of it (since all snippets would have those in common). Furthermore, ignoring words like *wikipedia*, *encyclopedia*, *dictionary* etc. made sense, as two different Wikipedia articles likely correspond to different query senses. The BoW also includes phrases in the form adjective_noun (as seen in [2, Section 3.1]). Lastly any and all non-alphabetic strings were ignored, too. This included URLs, cardinals, HTML formatting etc. The usefullness of all these filters is explored and tested in Section 3.2.1.

[1] `https://www.cs.york.ac.uk/semeval-2013/task11/index.php%3Fid=task-description.html`

2.2 Clustering

I chose to cluster the snippets according to the word overlap between their bag-of-words representations. However, a normalized overlap seemed more reasonable in order to take into account the power of the compared bags-of-words:

$$OL_{norm} = \frac{|B_1 \cap B_2|}{max(|B_1|, |B_2|)} \tag{1}$$

For example, an overlap of 3 between two bags-of-words of sizes 10 and 20 would lead to a normalized overlap of $3/20 = 0.15$, and the same overlap between BoWs of sizes 4 and 6 would achieve a normalized overlap of $3/6 = 0.5$, which is, of course, more significant.

The clustering algorithm starts with every snippet in its own cluster. It then finds the two most similar snippets s_i and s_j (according to the normalized overlap between their bags-of-words), and creates a new "snippet" in the form $(s_i\_id, s_j\_id) : B_i \cup B_j$. The initial two snippet IDs s_i and s_j are removed from the clustering. If there is more than one pair of "most similar" snippets, the algorithm chooses to merge the first pair it found. This process repeats while the normalized overlap between the two most similar snippets is higher than a certain overlap threshold λ. Tuning this threshold is explored in Section 3.2.2.

3 Development

3.1 Dataset

During the development of the system, I used the trial data set provided by the organizers. Since it only contains 4 queries, any results obtained by the system on this dataset might need to be taken with a grain of salt.

	# of queries	av. # of clusters in STRel.txt	snippets per query
TRIAL	4	2.75	100

Table 1: Trial dataset statistics.

3.2 Experiments

3.2.1 The BoW-extraction

Below are the different approaches to the BoW-representation I tried.

- **'phrases'** refers to the adjective_noun phrases mentioned in Section 2.1. 'full - phrases' means no phrases are included in the BoW representations of the snippets.
- **'is_alpha', 'is_stop'** are attributes of spaCy's `Token` class, checking whether a Token object consists of only alphabetic characters, and whether it is a stop word according to spaCy's list of stopwords. As filters, only nouns for which is_alpha returns True and is_stop returns False are included in the bag-of-words.
- **'special_stop'** refers to the list of special stopwords like wikipedia, encyclopedia etc., which should be excluded from the BoW.

- The **'full'** BoW-extraction system includes all the filters above, and additionally checks whether the noun or phrase are the query itself or part of it. 'full - X' is an ablation of the full system, in which the check/filter X is left out.

overlap: 0.02	TASK-trial				
	RI	ARI	JI	F1	Average
full	58.93	3.65	54.03	67.14	45.94
full - phrases	59.19	1.98	54.98	**67.35**	45.88
full - is_alpha	58.73	-0.33	52.28	66.74	44.35
full - is_stop	55.96	-1.16	52.20	66.19	43.29
full - special_stop	57.34	-2.10	53.54	65.92	43.67
full - length(> 3)	**60.29**	**4.84**	**55.43**	66.63	**46.80**
full - length - is_stop	58.18	1.83	54.02	66.07	45.03
full - length - phrases	58.41	1.53	54.10	66.42	45.12

Table 2: BoW systems.

The overlap threshold was fixed at 0.02 for these experiments. The clustering quality results were calculated with the evaluator provided by the task organizers. Di Marco and Navigli [2] include a detailed description of the used measures.

It is clear which filters have the strongest impact on the system scores. Removing *only* the special_stopwords check leads to a 5.7% drop in ARI score, and an overall drop of 2.3% in the averaged score of all four metrics. In fact, the only ablation that achieves an overall performance gain is the removal of the *length* > 3 check, which beats the 'full' system in all measures save F1.

Taking these results into account, I decided to use the 'full - length' BoW-extraction approach for the overlap experiments, and in the final system.

3.2.2 The Overlap Threshold

By observing the scores in Table 3, it is easy to draw the conclusion that the overlap threshold λ determines the average number of clusters produced by the system. This is due to the nature of the overlap threshold. A higher threshold causes the clustering algorithm to terminate earlier, thus limiting the number of merges between snippet BoWs. A lower λ, allows the merging to continue, resulting in a smaller number of clusters. The best overlap threshold for this dataset is 0.005, with an average of 50.2% over the four metrics. The average number of produced clusters is 2.25, which is close to the average number of clusters in the gold annotation (the file STRel.txt) - 2.75. Applying an overlap threshold this low to unseen data, however, might overgeneralize, i.e. continue merging clusters that should be separate. Accordung to Navigli and Vannella [1], the average number of senses associated with the search results of each query is 5.07 for the AMBIENT+MORESQUE[2] dataset (which our trial dataset is part of), and 7.69 for the new dataset presented in their paper. This information led me to choose an overlap threshold of 0.02 for my final system. While still yielding acceptable results on the

[2] `http://lcl.uniroma1.it/moresque`

λ	RI	ARI	JI	F1	# cl.
0	**68.38**	1.82	**67.46**	66.31	1.25
0.005	65.41	**6.47**	63.28	66.75	2.25
0.01	63.03	3.50	60.61	66.71	2.75
0.015	61.53	4.27	58.04	66.63	3.50
0.02	60.29	4.84	55.44	66.63	4.25
0.025	53.19	-1.16	45.08	67.48	6.00
0.03	52.74	-1.20	43.53	67.48	6.75
0.035	50.29	1.88	39.29	67.52	8.25
0.04	48.02	-0.36	35.14	67.76	9.50
0.045	46.17	0.31	30.86	68.25	11.50
0.05	45.92	1.17	28.46	**68.81**	13.25

Table 3: Overlap threshold

trial dataset, it should also allow relatively good performance on other datasets, e.g. the ones mentioned above.

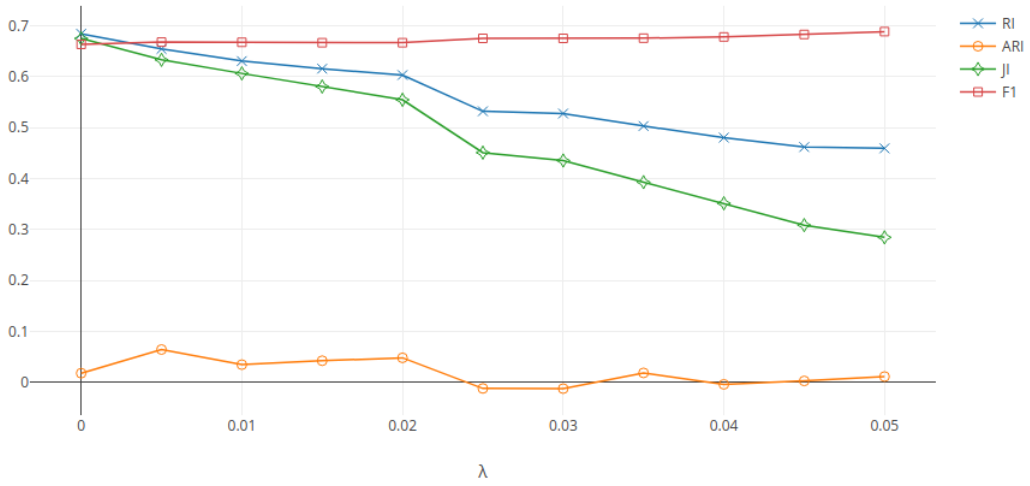

Figure 1: A graphic of the values in Table 3. The overlap threshold (λ) chosen for the final system is 0.02.

3.3 Comparison with other systems

I implemented the following baselines to see how well my final system performs:

- **Simplified** A version of my system with a different preprocessing component, using the TreeTagger tool[3], which greatly reduces runtime. Only single tokens were extracted, no phrases. The overlap threshold is fixed to 0, so the clustering algorithm terminates once the clusters have nothing more in common.

[3] `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`

- **All-in-One** All snippets appear in the output file, tagged with the same cluster ID.
- **Singletons** The output file is empty, save for the file header.

	RI	ARI	JI	F1
Final System	60.29	**4.84**	55.43	**66.63**
Simplified	68.71	1.01	67.69	66.05
All-in-one	**69.28**	0.00	**69.28**	66.10
Singletons	**69.28**	0.00	**69.28**	0.00

Table 4: Comparison

4 Discussion

The results are far from dazzling. At first glance the All-in-one and Singletons baselines seem impossible to beat in both Rand Index and Jaccard Index. The F1 measure scores tell me that my system prefers, and tends to produce large clusters. This is true particularly for the simplified version, since it generated an average of 1.25 clusters per query on the trial dataset, because of the fixed $\lambda = 0$ threshold. As a whole, my system produces one rather large cluster and (depending on the overlap threshold) several small ones (up to 5 snippets) and singletons. This is due to the clustering algorithm I used, and especially the merging of the bags-of-words of the most similar snippets. The new entry $(s_i\_id, s_j\_id) : B_i \cup B_j$ has as value a union of the BoWs, which in turn leads to a higher chance of overlap with other snippets, as the new set of words is larger. However, producing one large cluster and several small ones models the actual distribution of query senses in the search result snippets rather well. From the analysis of the STRel.txt file of this (limited) dataset, we can see that there is always one large(r) cluster, generally at least four times as large as the next largest. That is, one sense of the ambiguous query is more common than the rest.

# snippets per cluster	Query			
	soul food	stephen king	the block	cool water
Cluster 1	19	99	78	21
Cluster 2	5	-	8	2
Cluster 3	3	-	6	2
Cluster 4	-	-	2	-
Singletons	73	1	6	75

Table 5: Gold clusters

The most difficult metric is by far the Adjusted Rand Index, which takes into account the expected Rand Index value, and the actual value achieved. I was surprised to see that the systems compared in Navigli and Vannella's paper [1] also don't achieve an ARI score of more than 22% on their dataset. Actually, very few of the systems reached more than 7-8%. A source of improvent for any system would be figuring out a way to maximize its ARI score.

The Simplified system also performs better than my final system with respect to both RI and JI, but lags behind in ARI and F1. It is worth noting that even the best scoring system ($\lambda = 0.005$, Table 3) still fails to overcome Simplified in Rand Index and Jaccard Index scores.

5 Conclusion

I strived for simplicity with my idea and implementation, and cannot say that I am surprised by the results. I avoided too complicated approaches and focused on what should, intuitively, make sense and work well. It will be interesting to see how well my system performs on the test data, since I deliberately assumed an overlap threshold for my final system that didn't achieve the best possible results on the trial data.

Generally, I believe that bag-of-words representations are quite informative and useful in this particular task. They definitely scored better than the vector representations I tried out in the beginning. This is likely due to the possibility to tune the preprocessing steps, just by observing your output - something impossible when words and snippets feature in the system only as vectors.

This project was a new and useful experience for me. Working on it taught me the importance of knowing your data, which is essential for building a well-scoring system.

References

[1] R. Navigli, D. Vannella *SemEval-2013 Task 11: Word Sense Induction & Disambiguation within an End-User Application*. Seventh International Workshop on Semantic Evaluation (SemEval 2013) `http://www.aclweb.org/anthology/S13-2035`

[2] A. Di Marco, R. Navigli. *Clustering and Diversifying Web Search Results with Graph-Based Word Sense Induction*. Computational Linguistics, 39(4), MIT Press, 2013.