Neural Embedding Language Models in Semantic Clustering of Web Search Results

Andrey Kutuzov, Elizaveta Kuzmenko
University of Oslo, National Research University Higher School of Economics
andreku@ifi.uio.no, eakuzmenko_2@edu.hse.ru

Abstract

In this paper, a new approach towards semantic clustering of the results of ambiguous search queries is presented. We propose using distributed vector representations of words trained with the help of prediction-based neural embedding models to detect senses of search queries and to cluster search engine results page according to these senses. The words from titles and snippets together with semantic relationships between them form a graph, which is further partitioned into components related to different query senses. This approach to search engine results clustering is evaluated against a new manually annotated evaluation data set of Russian search queries. We show that in the task of semantically clustering search results, prediction-based models slightly but stably outperform traditional count-based ones, with the same training corpora.

Keywords: distributional semantics, neural embeddings, search results clustering

1. Introduction

Many user queries in web search are semantically ambiguous. For instance, the query *B-52* can relate to a bomber, a hairstyle, a cocktail or a rock band. Thus, search engines strive to diversify their output and to present results that are related to as many query senses as possible.

These results are as a rule not sorted by their relation to different ambiguous query senses and returned in the order of their relevance ranking, which for many of them seems to be almost equal. However, for the users' convenience in the case of an ambiguous query it is often plausible to consolidate results related to one meaning. It is especially important in vertical or personalized search or when mixing results from different search engines into one stream. The present paper deals with the problem of semantic clustering of a search engine results page (SERP), on the Russian language data.

Research related to search results clustering (SRC) has a long history: arguably, main milestones in the last decade are (Bernardini et al., 2009) and (Marco and Navigli, 2013). We follow the latter's graph-based approach, but augment it with word embedding algorithms, which has recently gained considerable traction in NLP. The formal description of the problem is: given an ambiguous search query and n top search results, cluster these results according to the corresponding query senses. Note that we deal only with the problem of clustering itself, leaving ranking results within cluster and labeling clusters with human-readable labels for further research. It is also important to emphasize that we do not use full document texts: our only input is their titles and snippets as presented on the search engine results page. Our main contributions are:

1. releasing a publicly available human-annotated dataset[1] for evaluation of semantic clustering for Russian search engine results, complementing existing data sets for English;
2. evaluating the performance of prediction-based distributional semantic models versus traditional count-based models for this task, using query graph approach.

The paper is structured as follows. In Section 2. we introduce our approach and put it into the context of the previous work in the field. In Section 3. the evaluation experiment is described and the results are discussed. In Section 4. we conclude and outline the future work.

2. Ways to cluster: drawing edges in the query graph

A straightforward approach towards semantic clustering of a search engine results page (SERP) is to rely on common words found in each result (a snippet and a title). Intuitively, the results which share words should belong to the same query sense. However, there are many cases when search snippets related to one sense contain completely different contexts and therefore do not share any words in common. Vice versa, results belonging to different senses can overlap lexically (by chance or because of some frequent words). Thus, in order to successfully cluster search results, the semantic relatedness or similarity of the words should be taken into account: the algorithm must 'understand' which words are semantically close to each other, even if their surface forms differ. Such data can be derived either from large ontologies or from word distribution in large text corpora. In this paper, we investigate the latter approach; cf. the well-known idea in general linguistics that word contexts determine the meaning of a word (Firth, 1957; Harris, 1970).

One possible way of applying this idea to SERP clustering suggests building a '**query graph**' G_q from words found in snippets and titles (Marco and Navigli, 2013). In this graph, the words are vertexes, with edges drawn between 'semantically similar' words. Additionally, words semantically similar to the query but absent from the SERP, are added to the graph. It is then partitioned in order to separate query senses, and each result is mapped onto the produced senses. Semantic similarity of the words is calculated on the basis of a lexical co-occurrence matrix built from a large reference corpus. Two words are considered

[1] http://ling.go.mail.ru/ruambient

similar if their distributional patterns are similar: that is, if they frequently occur near the same words in the reference corpus, according to some established collocation measure (Dice coefficient, point-wise mutual information, etc.).

We propose to calculate semantic similarity using distributed neural embedding models, instead of deriving it from a co-occurrence matrix directly (the latter is a so-called count-based method). In particular, we test distributed representations of words produced by the Continuous SkipGram algorithm, first implemented in the *word2vec* tool (Mikolov et al., 2013). We hypothesize that it should result in higher accuracy when determining semantic relations between words: it was previously shown that the prediction-based distributional semantic models outperform the count-based ones in semantic tasks (Baroni et al., 2014). Note, however, that it was also reported that a large part of their performance is determined by the choice of hyperparameters, not the algorithms themselves (Levy et al., 2015).

The most obvious way to implement distributional semantic models into graph-based semantic clustering implies using the cosine distance between word vectors in the trained model as a feature determining whether an edge should be added between the vertexes corresponding to these words in the query graph. However, this approach is highly unstable and dependent on the chosen similarity threshold: for different words, any given threshold would produce thousands of 'related words' in some cases and only a few in others. This is because cosine distance is not an absolute metrics, but a relative one: it makes sense only when we compare several values to determine which word pair is 'semantically closer' than the other ones.

That's why we use a slightly different approach to construct edges in the query graph. For every pair of vertexes V_i and V_j on the graph G_q, we extract from the trained model n most semantically similar lexical units for each of the two words corresponding to these vertexes. If either V_i is found in these 'nearest neighbors' of V_j, or vice versa, an edge is added between the vertexes, with the cosine distance between them serving as weight on this edge. Note that in production setting the query graphs for frequent queries can be cached to ensure scalability of the approach.

The produced graph is then partitioned into disconnected components. The resulting components approximate query senses, with vertexes in the components representing lexical inventory of the corresponding senses. Then, each result in the SERP is mapped onto one of these senses, using a simple token overlap measure.

3. Evaluation setup

Evaluation of SERP semantic clustering demands manually clustered data. For this, we produced the **RuAmbiEnt** (*Russian Ambiguous Entries*) dataset consisting of SERPs for 96 ambiguous single-word queries in Russian. The queries were taken from the *Analyzethis* homonymous queries analyzer[2]. *Analyzethis* is an independent search engines evaluation initiative for Russian, offering various search performance analyzers, including the one for ambiguous or homonymous queries. We crawled one of the major Russian search engines for these queries, getting titles and snippets of 30 top results for each query, 2880 results total.

This data was annotated by two independent human annotators manually mapping each result onto a corresponding query sense (subtopic). The sense inventory for the queries was produced beforehand by the third independent human annotator. The Adjusted Rand Index (Hubert and Arabie, 1985) between the two annotations is 0.94, proving that humans mostly agree on clustering of web search results. This dataset is the only one of this kind for Russian known to us. It follows the format used in AMBIENT (Bernardini et al., 2009) and MORESQUE (Navigli and Crisafulli, 2010) datasets for English, and is available for downloading at `http://ling.go.mail.ru/ruambient/`.

RuAmbiEnt serves as a golden standard for our system evaluation. Using the same Adjusted Rand Index (ARI), we measured the similarity of neural embedding based semantic clustering to human judgments and compared it to the performance of the traditional count-based approach described in (Marco and Navigli, 2013).

Count-based and prediction-based models were constructed in such a way that they were directly comparable. Both were produced from the same training corpus (the Russian National Corpus[3] augmented with the Russian Wikipedia dump from February 2015), containing 280 million word tokens after stop-words removal. The corpus was lemmatized using the Mystem tagger (Segalovich, 2003).

Continuous Skipgram models were trained using the Gensim library (Řehůřek and Sojka, 2010). Count-based models were based on constructing Positive Pointwise Mutual Information (PPMI) matrices between all words in the corpus. We employed PPMI as a collocation strength measure to make the comparison more fair: it was previously shown that the neural word embedding algorithms introduced in (Mikolov et al., 2013) implicitly factorize word-context matrix with PMI values in the cells (Levy and Goldberg, 2014). So, we reduced the dimensionality of the resulting matrices to 300 components, using SVD. The same vector size (300) was set during the training of SkipGram models.

To control all the other hyperparameters, we built count-based models using the *Hyperwords* framework (Levy et al., 2015). It allows to tune all the important settings so that the difference in performance could be attributed only to the difference in algorithms themselves, not to some peculiarities in preprocessing.

Thus, both types of models used symmetric window of 1 (only immediate left and right neighbors of words were considered) and no downsampling. Removal of rare words mimicked default behavior of reference Continuous SkipGram implementation (*word2vec*): that is, after removal, the window changes accordingly (the word actually acquires new neighbors). We tried two versions of the corpus with different frequency thresholds: in the first version we removed all *hapax legomena* (words with frequency equal

[2] `http://analyzethis.ru/?analyzer=homonymous`

[3] `http://ruscorpora.ru/en`

Table 1: Models performance on RUSSE semantic similarity dataset (Spearman's ρ for *hj*, average precision for the others)

	PPMI+SVD		**SkipGram**	
Frequency threshold	5	2	5	2
hj track	0.49	0.48	0.49	0.49
rt track	0.76	0.75	0.76	0.75
ae track	0.79	0.79	0.79	0.79
ae2 track	0.77	0.77	0.77	0.77

to 1), while in the second version we removed all the words with frequency less than 5.

Further, when calculating PPMI, context distribution smoothing of 0.75 was used, following the observation that it mimics sampling negative contexts from smoothed distribution in *word2vec* implementations and helps to alleviate PMI bias towards rare words (Levy et al., 2015). We used negative sampling of 10 to train the Continuous SkipGram models, and the same value of 10 was used to shift PPMI in the count-based models. Neural models were trained in 5 iterations over the corpus; for count-based models iterations parameter does not make sense, as they process the whole co-occurrence matrix at once, without any non-deterministic 'training' (in the case of Continuous SkipGram, the training is performed using stochastic gradient descent).

After these steps, both approaches provided us with lists of 300-dimensional vectors for all the words in the corpus, so that semantic similarity between words can be calculated as cosine distance between their corresponding vectors. For reference, in Table 1 we report the performance of the resulting models in a semantic similarity task, as evaluated against RUSSE dataset (Panchenko et al., 2015). There are four tracks in RUSSE evaluation framework, with *hj* measuring correlation with human judgment experiment, *rt* using a semantic thesaurus created by linguists, and *ae* and *ae2* measuring associative relations based on different experiments.

Note that all the models demonstrate similar performance in this evaluation task. This is not surprising, given identical training corpus and our efforts to use comparable preprocessing workflow. It supports the opinion of (Levy et al., 2015) that with the right combination of hyperparameters, PPMI+SVD performs on par with Continuous SkipGram in standard evaluation frameworks. However, as we show below, their behavior differs when applying them to practical tasks, like SERP semantic clustering.

The query graphs themselves were constructed closely following the work-flow described in (Marco and Navigli, 2013). Additionally, we experimented with several hyperparameters of query graph construction in order to find the optimal settings. These parameters included:

1. the number of nearest neighbors n to consider when drawing an edge; an edge is drawn between vertexes V_i and V_j only if at least one of them is among the n nearest neighbors of another in the current model;
2. threshold of cosine distance c; edges with weights lower than c are never added to the graph, thus, low-frequency noise from 'semantically isolated' words is removed.

Table 2: The ARI performance of the count-based models and the Skip-Gram word embedding algorithm

	10 results	30 results
Single-sense baseline	0.58	0.54
Best PPMI+SVD model	0.58	0.54
Best Skip-Gram model	**0.60**	**0.57**

Table 2 lists the ARI of our best-performing models in comparison to human judgments for full dataset (30 results) and for the first 10 results for each query. This latter version of the dataset was analyzed as it closely resembles real first page of search results in commercial search engines. We also report the performance of the single-sense baseline: an approach when all the results are clustered into one meaning.

It can be seen that the single-sense baseline is tough: usually the majority of the results for an ambiguous query tend to belong to one query sense (most frequent one). Thus it is easy to closely imitate human judgment by simply merging all the results into one cluster. Best-performing count-based model in our experiment was not able to overcome this baseline.

However, if the same graph-based algorithm is fed with word similarity data produced from Skip-Gram model (trained on the same corpus), the results get better. It is visible both with the full dataset and with the limited 10-results version. The difference does not seem large, but as we use the adjusted version of Rand Index as a measurement, we can be sure that it is statistically significant.

It is interesting that for all the models the full dataset presented a more difficult task: it seems that the second and the third pages of SERP contain more bad snippets and irrelevant results, which leads to worse graph partitioning.

Another important finding is that both prediction-based models and the traditional count-based PPMI+SVD approach do not benefit from additional elaborate graph partitioning algorithms, like *Curvature* (Marco and Navigli, 2013) or *Hyperlex* (Véronis, 2004). In most cases, the query graph produced from the SERP already consists of several disconnected components[4] often matching query senses: see the example on the Figure 1, where the senses of '*animal*' (red giant component), '*fitness club*' (green vertexes) and '*consumer electronics shop*' (blue vertexes) of the word 'zebra' are in different graph components. Triangular vertexes reflect words which were added from query word nearest neighbors in the model, not from SERP.

We suppose the reason for this behavior is that we used more balanced and clean training text collections, unlike web-derived corpora employed in (Marco and Navigli, 2013), where partitioning was necessary to extract mean-

[4]If it doesn't (which is sometimes the case for PPMI+SVD), using additional graph partitioning does not change this.

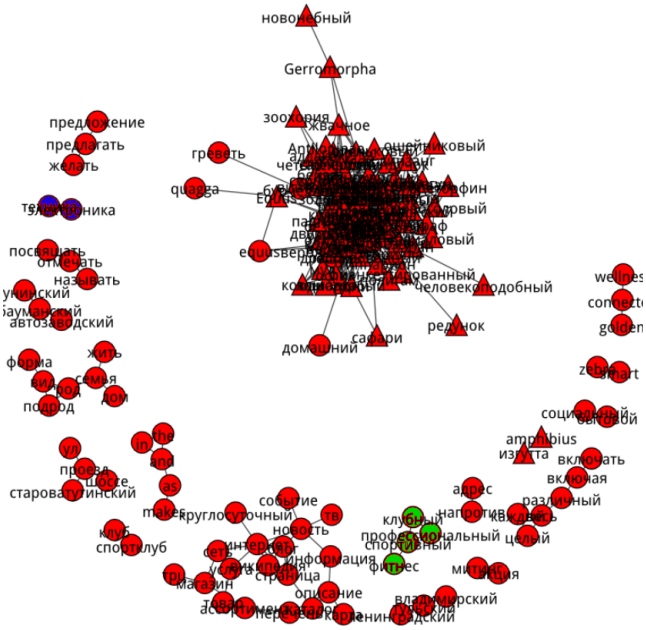

Figure 1: Query graph for ‘зебра’ (‘zebra’), produced on Continuous SkipGram model

ingful clusters from the query graph. Another possible explanation is the influence of language-specific factors; experiments with English material are needed to verify this.

Considering empirically determined hyperparameters of our algorithms, in the best-performing models of both approaches, n was 100 (we draw an edge if V_i is in 100 nearest neighbors of V_j or vice versa), and c was 0.1 (if similarity between V_i and V_j is less than 0.1, we do not draw an edge). Interestingly, for Continuous SkipGram the model using corpora with frequency threshold 2 turned out to be the best, while for PPMI+SVD it was the model with the higher frequency threshold 5. It seems that in spite of all efforts to cope with PPMI bias towards infrequent entities, it still negatively influences the results. This is why count-based models perform better in this task when frequency threshold is high: low-frequency words are filtered out and thus there is less noise. At the same time, this is not an issue for prediction-based models: decreasing frequency threshold for them results in better performance. It means that at least with SERP semantic clustering neural embedding models are able to employ the knowledge contained in low-frequency co-occurrences significantly better than count models like PPMI+SVD.

The best-performing Continuous SkipGram model and the Python code to employ it in SERP clustering problem are available online[5].

4. Conclusion

In this paper we compared traditional count-based distributional semantic models and prediction models based on neural word embeddings (the Continuous Skip-Gram algorithm) for a particular task of search results clustering for Russian ambiguous queries. Our experiments show that for this task, neural embedding models indeed perform stably better in identical settings, though the difference is not striking.

Maximum ARI in comparison with human judgment reached by count-based models was **0.58**, equal to the performance of the simple single-sense baseline. At the same time, prediction-based models were able to reach ARI of **0.6**.

In the future, we plan to refine the methodology of graph-based semantic clustering. It seems promising to experiment with other partitioning algorithms. With distributed models, the query graph is often divided into disconnected components ‘out of the box’ (see Section 3.), but advanced algorithms such as *Chinese whispers* (Biemann, 2006) can still potentially improve clustering quality.

Also, we are in the process of evaluating our approach on available English datasets and models to find out whether the results are consistent across languages.

[5] https://cloud.mail.ru/public/A8AW/jAA2LaGdx

5. Bibliographical References

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don’t count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1.

Bernardini, A., Carpineto, C., and Amico, M. D. (2009). Full-subtopic retrieval with keyphrase-based search results clustering. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT’09. IEEE/WIC/ACM International Joint Conferences on*, volume 1, pages 206–213. IET.

Biemann, C. (2006). Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the first workshop on graph based methods for natural language processing*, pages 73–80. Association for Computational Linguistics.

Firth, J. (1957). *A synopsis of linguistic theory, 1930-1955*. Blackwell.

Harris, Z. S. (1970). *Distributional structure*. Springer.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218.

Levy, O. and Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In *Advances in Neural Information Processing Systems*, pages 2177–2185.

Levy, O., Goldberg, Y., and Dagan, I. (2015). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Marco, A. D. and Navigli, R. (2013). Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Navigli, R. and Crisafulli, G. (2010). Inducing word senses to improve web search result clustering. In *Proceedings*

of the 2010 conference on empirical methods in natural language processing, pages 116–126. Association for Computational Linguistics.

Panchenko, A., Loukachevitch, N., Ustalov, D., Paperno, D., Meyer, C., and Konstantinova, N. (2015). Russe: The first workshop on russian semantic similarity. *Computational Linguistics and Intellectual Technologies: Papers from the Annual Conference. Dialogue*, 2:89–105.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May.

Segalovich, I. (2003). A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine. In *MLMTA*, pages 273–280. Citeseer.

Véronis, J. (2004). Hyperlex: lexical cartography for information retrieval. *Computer Speech & Language*, 18(3):223–252.