TUE at SemEval-2020 Task 1: Detecting semantic change by clustering contextual word embeddings

Anna Karnysheva
University of Tübingen
`anna.karnysheva@student.uni-tuebingen.de`

Pia Schwarz
University of Tübingen
`pi.schwarz@student.uni-tuebingen.de`

Abstract

This paper describes our system for SemEval 2020 Task 1: *Unsupervised Lexical Semantic Change Detection*. Target words of corpora from two different time periods are classified according to their semantic change. The languages covered are English, German, Latin, and Swedish. We use ELMo embeddings and cluster the embeddings with DBSCAN and K-means. For a more fine grained detection of semantic change we take the Jensen-Shannon Distance metric and rank the target words from strongest to weakest change. The results show that this is a valid approach for the classification subtask where we rank 13th out of 33 groups with an accuracy score of 61.2%. For the ranking subtask we score a Spearman's rank-order correlation coefficient of 0.087 which places us on rank 29.

1 Introduction

Although lexical semantic change has been a topic of interest for historical linguists for well over a hundred years, the technological advances have provided new computational methods to tackle this problem. As a result, numerous systems have been developed for a range of different languages and different time periods, using different evaluation metrics (for a comprehensive survey see Tahmasebi et al. (2018)). However, there is no universal evaluation framework, which makes the assessment across different approaches difficult. Diachronic lexical semantic change detection can be helpful to simplify the understanding and research with archived historical data which becomes accessible with the progress of digitization.

SemEval 2020 Task 1 is an unsupervised lexical semantic change detection task, consisting of two subtasks - a classification and a ranking subtask covering English, German, Latin and Swedish (Schlechtweg et al., 2020). Given two corpora from two different time periods, in subtask 1 individual target words have to be classified into the binary categories of *sense change* or *no sense change*, where the class *sense change* does not distinguish between a word losing or gaining sense(s). An example is the English word "bit" which over time gained the sense of a *computational unit*, additionally to the meaning of *small quantity*. Subtask 2 consists in ranking the target words according to the degree of lexical semantic change. The word with the strongest change is assigned the highest rank.

Our approach is based on clustering contextual word embeddings with two different algorithms – K-means and DBSCAN. We avoid embedding space alignment by combining both, corpus C1 from time period t_1 and corpus C2 from time period t_2. The number and size of the clusters determine whether a word exhibits sense change or not. We calculate the degree of change of a word with the Jensen-Shannon distance (JSD) (Lin, 1991) with probability arrays of the cluster distributions.[1]

[1]The source code of our model is published at `https://github.com/basagashka/SemEval2020-Task1-Schwaebischschwaetza_tue`.

Language	**C1 time period**	**C2 time period**	k	n	**target words**	**size C1**	**size C2**
ENG	1810 - 1860	1960 - 2010	2	5	37	6.6	6.8
GER	1800 - 1899	1946 - 1990	2	5	48	70.2	72.4
LAT	200 BCE - 0	0 - 2000	0	1	40	1.8	9.4
SWE	1790 - 1830	1895 - 1903	2	5	31	71.1	110.8

Table 1: Corpora time periods, thresholds k and n for classification, number of target words, and approx. corpus word counts (in million) per language.

2 Task Description

For this task we were provided with corpus C1 and corpus C2, where the time period t_1 of C1 lies before the time period t_2 of C2, and a target word list for English, German, Latin and Swedish (for details about the corpora see Schlechtweg et al. (2020)). The time periods covered in the corpora differ in their span as well as in beginning and end dates. While there is no break between the time periods of t_1 and t_2 of the Latin corpora, all other languages have breaks ranging from 47 to 100 years. In addition, whereas the second Latin corpus covers a time period of 2000 years, all other C1 and C2 time periods do not exceed 200 years. All corpora were lemmatized, shuffled and formatted to one sentence per line. The target word lists, consisting of nouns and verbs, are of different sizes: with German and Latin being the longest ones (48 and 40 target words respectively), followed by English with 37 words and Swedish with 31. The word count in the test corpora for the system evaluation ranges from 1.7 to 110 million words.

Given the corpora C1 and C2, each target word has to be classified into two categories for subtask 1. Between the time period of C1 and the time period of C2 a target word either (i) exhibits semantic change, meaning it has lost or gained one or several sense(s), or (ii) there has been no semantic change. For a word to have gained a sense, the sense has to appear a maximum of k times in C1 and at least n times in C2. For a word to have lost a semantic sense, the sense has to appear a maximum of k times in C2 and at least n times in C1. The values for k and n were set differently for Latin ($k = 0$, $n = 1$) as it has a smaller sample size than the other three languages ($k = 2$, $n = 5$).

For example, assuming the target word "bit" has two senses, the first one referring to a *small quantity* and the other one to the *computational unit*, and the sense distribution of the meanings in C1 and C2 is as described in Table 2. It is evident that the target word "bit" has acquired the sense *computational unit*, as this sense was detected at most 2 times ($k \leq 2$) in C1 and at least 5 times ($k \geq 5$) in C2; "bit" is therefore classified as (i) exhibiting semantic change.

senses *bit*	**C1**	**C2**
small quantity	25	16
computational unit	0	8

Table 2: Word sense distribution across C1 and C2 for the target word *bit* (exemplary numbers).

The goal of subtask 2 is to determine the degree of lexical semantic change of each target word and then rank all target words according to this degree, where the item with the highest degree of semantic change is ranked first. Using the Jensen-Shannon Distance metric the word sense frequency distributions of "bit" has a result bound between 0 and 1. Whereas the frequency of the first sense of *small quantity* is decreasing, it is increasing in the second sense of *computational unit*. The degree of lexical semantic change can capture a semantic change which is less coarse than the classification scores from subtask 1 suggest.

3 System overview

The key idea of our approach is to use contextualized embeddings to embed the target words and then cluster the resulting vectors. Word embeddings are dense vector representations that encode context information. According to the famous principle of Firth (1957) the meaning of a word depends on its surrounding words, i. e. its context. While several architectures of contextual embeddings exist, we make use of the ELMo architecture. As our primary task in subtask 1 lies in being able to track if a sense changes or not, we need to be able to differentiate between different senses of a word. Since ELMo uses all the internal representation layers to compute the final representation, it is supposed to capture more information at different linguistic levels (semantics, syntax, etc.) than simple word embeddings (Peters et al., 2018).

A shortcoming of pretrained embeddings is that they are trained on non-lemmatized corpora, unlike the corpora provided by this task which are lemmatized. This might have an influence on the quality of the embeddings we retrieve.

3.1 Preprocessing

Before computing the word embeddings, C1 and C2 of each language are combined. This allows for a single embedding space which lets us skip the alignment step that would be necessary if target words in C1 and C2 were embedded in separate spaces. In order to be able to tell whether a target word belongs to a sentence from C1 or C2, we collect the indices of every word (sentence and word position) in the corpora as well as the sentence index of the boundary where C1 ends and C2 starts.

3.2 Embedding and clustering

To retrieve the embeddings per target word, only sentences containing the target word are embedded. Finally, with the help of the previously collected target word indices, only the target word vectors are selected. The resulting vector coordinates (in this paper also referred to as data points) are then clustered in the next step.

We employ two different clustering algorithms: for English and Swedish we apply the K-means (Lloyd, 1982) algorithm, and for German and Latin we use density-based spatial clustering of applications with noise (DBSCAN) (Ester et al., 1996). Although we have tried applying both algorithms on all four languages, the combinations described above seem have yielded better results for the given task. The advantage of DBSCAN over K-means is that it does not require initialization with the number of clusters, which, in our case, is different for each target word. Moreover, the shape of the clusters has less of an impact on DBSCAN than on K-means. Although the advantages of DBSCAN are clear, there is also a downside to using it, as DBSCAN needs two hyperparameters: *minimum samples*, the minimum amount of data points for a sample to be considered a core sample, and *epsilon*, the maximum distance between these data points, while K-means settles for a single one.

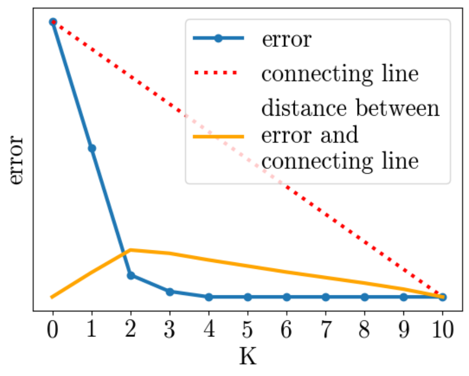

Figure 1: Scree plot with the “elbow” at $K = 2$

To address the issue of initializing K-means clustering with a value for the number of clusters we use a dynamic approach to find K automatically for each target word.[2] The possible values that K can take are set through a range. In our model, the upper bound of the range is 10, meaning that the K-means

[2]We adopted the idea from Bhavesh Bhatt, available at `https://www.youtube.com/watch?v=IEBsrUQ4eMc`.

clustering is ran with values from 1 to 10. K-means converges when the squared error of the distance between the data points and their cluster center reaches a local minimum. Plotting this value against the different values of K results in a scree plot which usually depicts a line with an "elbow" (circle marked line in Figure 1). To automatically find where the elbow lies, another line (dotted) connecting both ends of the squared error line is inserted. The maximum of the distance (peak of the solid line) from the dotted line to the line representing the squared error is exactly where the elbow is found. The connecting line always starts from the squared error where K is lowest, i.e. where $K = 1$; the distance will thus always be 0 there (the same applies to the distance at $K = 10$). To allow for a word to have a single cluster we therefore need to insert an artificial value for $K = 0$. This value for $K = 0$ is computed by multiplying the value for $K = 1$ with a factor t where $1 < t \leq 2$, leading to another hyperparameter that needs to be tuned.

3.3 Determining class and degree of change

The previous clustering step results in a list of labels which is split into two separate lists, the labels for C1 and C2. After counting the distinct labels and their number of occurrences in both lists, the target word is classified accordingly into (i) there is semantic change or (ii) there is no semantic change, taking into account the thresholds of k and n (see Table 1) from the specifications of subtask 1.

target word	**Clusters (label: occur.)**	**Probability distribution**
C1	0: 18, 1: 53, 2: 9	$[\frac{18}{80}, \frac{53}{80}, \frac{9}{80}]$
C2	0: 51, 1: 2	$[\frac{51}{53}, \frac{2}{53}, 0]$

Table 3: Cluster and probability distributions of a target word (exemplary numbers).

Based on these label lists for C1 and C2, we also compute the degree of semantic change required for subtask 2 by converting the cluster distributions into corresponding probability distributions as exemplified in Table 3. These probability distributions are used to calculate the degree of lexical semantic change using the SciPy implementation of the Jensen-Shannon Distance metric (Virtanen et al., 2020).

4 Experimental setup

For German, Latin and Swedish we employ *Pre-trained ELMo Representations for Many Languages* which are trained on a corpus of 20 million tokens. (Che et al., 2018).[3] Although there was an available English model as well, we decided to use *Deep contextualized word representations* for English, a model implementation of ELMo trained on a 1 billion word benchmark, primarily due to the substantially greater corpus size of its training data (Peters et al., 2018).[4]

In total, our system has three hyperparameters: *minimum samples* and *epsilon* for DBSCAN and t for K-means (Pedregosa et al., 2011) which we tune on two embedding settings.[5] While the first setting uses the target word embeddings, the second setting uses embeddings of the context around the target word (i.e., the sentence in which the target word appears, with the target word excluded). We only report results with target word embeddings due to substantially worse results with context embeddings during the tuning process. We did not have a development set available as the target word predictions of the trial data provided were randomly assigned, meaning that they do not reflect real values. Instead, we draw on the binary change score from a data set based on the SemCor corpus as development data to tune our English model (Schlechtweg and Schulte Im Walde, 2020). As the Latin corpora provided by the organizers contain homonym annotations, which indicate different senses of homonyms, we use those to train Latin.[6] For K-means, we obtain the highest accuracy scores with $t = 1.3$ for English and $t = 1.7$ for

[3]Available at `https://github.com/HIT-SCIR/ELMoForManyLangs`.

[4]Available at `https://tfhub.dev/google/elmo/3`.

[5]See Appendix for all other parameters for the DBSCAN and K-means algorithms.

[6]The reference for the annotation is the Lewis-Short dictionary, an example would be the word *dico*, available at `http://www.perseus.tufts.edu/hopper/morph?l=dico&la=la#lexicon`.

Latin. For DBSCAN tuning results in $epsilon = 5$, $min.samples = 2$ for English and $epsilon = 3.5$, $min.samples = 3$ for Latin. Due to the lack of development data for Swedish and German we use the settings from English for these two languages as they also belong to the Germanic language family. The combination of K-means and DBSCAN that yields the highest score during evaluation are K-means for English and Swedish and DBSCAN for German and Latin. The baselines for the tasks are the majority class baseline for subtask 1; subtask 2 has two baseline models: normalized frequency difference and count vectors with column intersection and cosine distance (Schlechtweg et al., 2020).

5 Results

For subtask 1, the results are evaluated against binary annotations done by humans. Our results for all four languages reach an average accuracy of 61.2%. This corresponds to an overall rank 13 out of 33 participating groups, where the accuracy lies between 41.3% and 68.7%. Our best individual accuracy score is the one for Latin with 65%.

	Accuracy	**SPR**
English	56.8%	-0.155
German	58.3%	**0.388**
Latin	**65%**	0.177
Swedish	64.5%	-0.062
Average	61.2%	0.087

Table 4: Results for subtask 1 (accuracy) and subtask 2 (SPR).

Subtask 2 is evaluated using the Spearman's rank-order correlation coefficient (SPR) against real human annotations as well. Our average score over all four languages is 0.087. Other participants' scores range from the best score of 0.527 to -0.083. Here, our best individual score is 0.388 for German.

Looking at the predictions our system outputs for English in subtask 1, we see that all target words are classified into the same class which might indicate an error in our system and is probably the reason for the low accuracy score of 56.8%. Moreover, our rather low scores for subtask 2 suggest that using the cluster distributions from subtask 1 without any further processing than using JSD is not ideal.

6 Conclusion

Although we show that tackling unsupervised detection of lexical semantic change by using well known clustering algorithms combined with contextual word embeddings is a valid approach, as our accuracy scores are above the score of the baseline model, our system is far from perfect, our results for subtask 2 point out the flaws of our system. A possible improvement of the current model would be to train ELMo on corpora that contain the target words instead of using pre-trained embeddings. This would also resolve the issue of using embeddings trained on unlemmatized data while the actual test data is lemmatized. Another aspect for future work might be experimenting with other clustering algorithms that account for more varied cluster shapes as for example DBSCAN does.

References

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium, October. Association for Computational Linguistics.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press.

J. R. Firth. 1957. A synopsis of linguistic theory 1930-55. 1952-59:1–32.

J. Lin. 1991. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151.

Stuart P. Lloyd. 1982. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Dominik Schlechtweg and Sabine Schulte Im Walde. 2020. Simulating lexical semantic change from sense-annotated data. In A. Ravignani, C. Barbieri, M. Martins, M. Flaherty, Y. Jadoul, E. Lattenkamp, H. Little, K. Mudd, and T. Verhoef, editors, *The Evolution of Language: Proceedings of the 13th International Conference (EvoLang13)*.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. Semeval-2020 task 1: Unsupervised lexical semantic change detection. *To appear in SemEval@COLING2020*.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to lexical semantic change.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Appendix

Parameter settings of DBSCAN and K-Means:

DBSCAN	**K-means**
metric=euclidean	init=k-means++
metric_params=None	n_init=10
algorithm=auto	max_iter=300
leaf_size=30	tol=1e-4
p=None	precompute_distances=auto
n_jobs=None	verbose=0
	random_state=None
	copy_x=True
	n_jobs=None
	algorithm=auto