CMCE at SemEval-2020 Task 1: Clustering on Manifolds of Contextualized Embeddings to Detect Historical Meaning Shifts

David Rother[1], **Thomas Haider**[2,3], **Steffen Eger**[1]
[1]Department of Computer Science, Technical University of Darmstadt
[2]Department of Language and Literature, Max Planck Institute for Empirical Aesthetics
[3]Institut für Maschinelle Sprachverarbeitung, University of Stuttgart
david.rother@stud.tu-darmstadt.de, thomas.haider@ae.mpg.de
eger@aiphes.tu-darmstadt.de

Abstract

This paper describes the system Clustering on Manifolds of Contextualized Embeddings (CMCE) submitted to the SemEval-2020 Task 1 on Unsupervised Lexical Semantic Change Detection. Subtask 1 asks to identify whether or not a word gained/lost a sense across two time periods. Subtask 2 is about computing a ranking of words according to the amount of change their senses underwent. Our system uses contextualized word embeddings from MBERT, whose dimensionality we reduce with an autoencoder and the UMAP algorithm, to be able to use a wider array of clustering algorithms that can automatically determine the number of clusters. We use Hierarchical Density Based Clustering (HDBSCAN) and compare it to Gaussian Mixture Models (GMMs) and other clustering algorithms. Remarkably, with only 10 dimensional MBERT embeddings (reduced from the original size of 768), our submitted model performs best on subtask 1 for English and ranks third in subtask 2 for English. In addition to describing our system, we discuss our hyperparameter configurations and examine why our system lags behind for the other languages involved in the shared task (German, Swedish, Latin). Our code is available at https://github.com/DavidRother/semeval2020-task1

1 Introduction

With the advent of increasingly sophisticated distributional models, semantic change analysis has gained considerable popularity in natural language processing. Earlier methods use static word embeddings to identify laws of semantic change (Hamilton et al., 2016a; Hamilton et al., 2016b; Eger and Mehler, 2016). Recently, contextualized embeddings (Devlin et al., 2018) have allowed to track more fine-grained sense changes of words over time (Hu et al., 2019).

The SemEval-2020 "Unsupervised Lexical Semantic Change Detection task" (Schlechtweg et al., 2020) aims at two levels of granularity to detect lexical semantic change: 1) a binary classification task to identify whether or not a word lost/gained a sense between two time periods, and 2) a ranking task, where a list of words has to be ordered by the amount of change these have undergone between the same two time periods. Both subtasks have to be solved for four languages: English, German, Latin and Swedish. To address the tasks, we deploy a language-agnostic system with three ingredients. i) We represent words by multilingual contextualized word embeddings (MBERT) (Devlin et al., 2018). ii) We apply dimensionality reduction on the contextualized word embeddings with an autoencoder and the UMAP algorithm (a type of manifold learning), as it is much easier and more efficient to work with low dimensional vectors in subsequent steps, allowing the application of a wider range of clustering algorithms. iii) We employ a hierarchical clustering approach to find potential 'sense clusters' for each word. These sense clusters can then be used to find sense losses or gains and shifts in the sense distribution between two time epochs.

While our system ranks in the mid-field overall, we perform best for English in subtask 1 and third in subtask 2. While we can only speculate about the approaches of the other participants, we believe this to be due to our language-agnostic approach, which uses no language-specific resources whatsoever except

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 187–193
Barcelona, Spain (Online), December 12, 2020.

for large-scale contextualized embeddings, possibly limiting its success in lower-resource languages such as Latin. Our analysis further suggests that hyperparameter tuning would have been crucial for better performances in Swedish and German.

2 Task Specification

	English	German	Swedish	Latin
Epoch t_1	1810-1860	1800-1860	1790-1830	-200-0
Epoch t_2	1960-2010	1946-1990	1895-1903	0-2000
#tokens t_1/t_2	6M / 6M	70M / 72M	71M / 111M	1.7M / 9.4M
#Focus words	37	48	31	40

Table 1: Corpus Overview Statistics

We aim to detect word sense change for given lemmatized focus words in English, German, Swedish and Latin. For each involved language, the task organizers supply a corpus for two epochs; statistics are given in Table 1. Note that the corpus sizes vary considerably across the languages, with Latin being smallest in size. The manual annotation of the test data against which all systems were eventually evaluated was done by human experts according to the guidelines by Schlechtweg et al. (2018). Annotators were asked to label focus words with one of several discrete senses. The precise definition of the subtasks of the shared task is the following:

- **Subtask 1** is a binary classification task to identify whether or not a word lost/gained a sense between two time periods t_1 and t_2, i.e., if it underwent meaning change or not. By the task organizers' specification, a sense counts as lost/gained if it occurs at least $n = 5$ times in t_1 (t_2) and at most $k = 2$ times in t_2 (t_1) (Latin had $n = 2$ and $k = 0$, respectively).
- **Subtask 2** asks to rank a list of words by the amount of change they have undergone between t_1 and t_2. For the test data, a ranking was determined according to the Jensen-Shannon Distance (JSD) between the distribution of senses between both epochs for each word. Ultimately, the fit of the system is computed by calculating the Spearman rank correlation between the model output and the gold ranking.

3 System Overview

To build a system that accomplishes these tasks, we use a pre-trained deep language model in order to have a meaningful and rich representation of words that is sensitive to the respective context. We then employ unsupervised clustering algorithms to combine the different representations and additionally try dimensionality reduction methods to boost clustering performance.

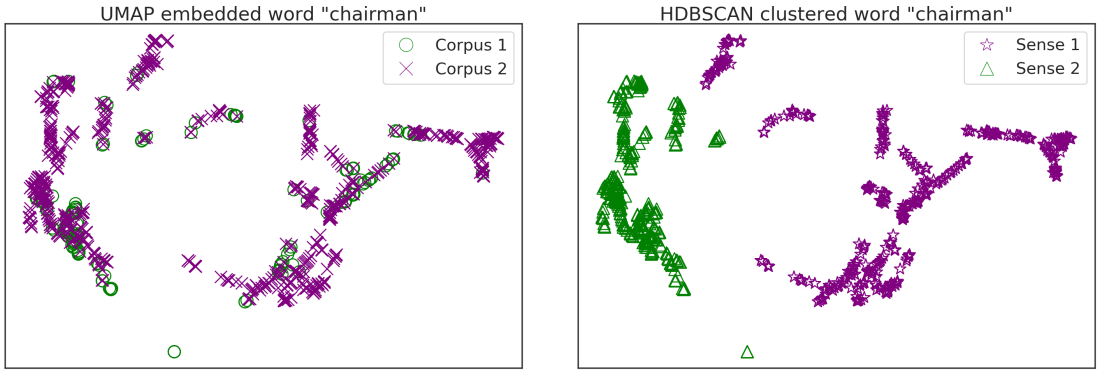
In our submission we use a pipeline that implements those steps. The language model we use is MBERT from which we retrieve contextualized word embeddings of the focus words from its last layer. We subsequently perform dimensionality reduction by first training and applying an AutoEncoder model on the data and then apply the UMAP algorithm afterwards. For the purpose of finding the different sense clusters we use Hierarchical Density Based Clustering (HDBSCAN) as a last step.

For illustration, we plot the instances for the word 'chairman' in Figure 1. Our clustering finds two distinct senses which corresponds to the annotation in WordNet (Fellbaum, 2012). The official competition scores of our submission are shown in Table 2, and Table 4 lists the hyperparameter settings of this submitted system. In Table 3 we report results with optimized hyperparameters on the post-evaluation data, comparing the clustering algorithms HDBSCAN and an alternative GMM approach. When the best configuration is known, we become more competitive overall (taking into account that the competing systems likely also improve when tuning their hyperparameters). In the following, we describe the steps of our pipeline. Further results from the post-evaluation are reported in the respective sections.

3.1 Contextualized Embeddings

First, we compute contextualized embedding vectors ($d = 768$) from MBERT's last layer for each focus word within its sentential context. Our intuition is that senses are represented by vectors with a small cosine distance to each other since they occur in similar contexts. We use the bert-base-multilingual-cased model of Huggingface (Wolf et al., 2019) that was trained on 104 languages and 110M parameters. MBERT is attractive because it a) is available for all shared task languages; b) can disambiguate on a sense level (Coenen et al., 2019); c) has shown strong performance across multiple NLP tasks. We perform no fine-tuning (e.g., on the historical data), as previous research found that this may lead to decreased performances in small data scenarios (Giulianelli, 2019).

Additionally, since the backbone of our pipeline are contextualized embeddings, one possible source for errors is the quality of the embeddings. We also test our system with XLM-R (Conneau et al., 2019) embeddings, but the final results of the system do not systematically improve.



(a) Visualization of the UMAP projected auto encoded MBERT embeddings of the word 'chairman' in two-dimensional space. Embeddings from both epochs are scattered throughout the same areas.

(b) Example clustering result of the HDBSCAN algorithm. Two distinct senses are detected based on connectivity and density of the two regions.

Figure 1: Two-dimensional visualization for the instances of the word 'chairman', according to (a) their time epochs and (b) the sense clusters as found by HDBSCAN.

		Ours	Best System
English	subtask 1	**.730**	**.730**
	subtask 2	.440	**.458**
German	subtask 1	.542	**.812**
	subtask 2	.412	**.735**
Swedish	subtask 1	.613	**.774**
	subtask 2	.114	**.604**
Latin	subtask 1	.450	**.725**
	subtask 2	.109	**.513**
All	subtask 1	.584	**.687**
	subtask 2	.269	**.527**

Table 2: Best results of our system vs. best performance of any system for subtask 1 (measured in accuracy) and subtask 2 (measured in Spearman's ρ).

		GMM	HDBSCAN
English	subtask 1	.622	.730
	subtask 2	.306	.512
German	subtask 1	.729	.625
	subtask 2	.605	.604
Swedish	subtask 1	.742	.806
	subtask 2	.268	.308
Latin	subtask 1	.575	.525
	subtask 2	.321	.227
All	subtask 1	.667	.671
	subtask 2	.375	.413

Table 3: Aggregated best results after hyper-parameter optimization on the test set (post-evaluation) for both subtasks per language for the clustering methods HDBSCAN and GMM. No model performs best across all languages individually.

	English	German	Swedish	Latin
UMAP: Number of neighbors	6	6	5	3
HDBSCAN: Cluster size ratio	.03	.06	.025	.01

Table 4: Final hyperparameter configuration for UMAP and HDBSCAN in the best run of the competition submissions: Number of neighbors for UMAP and the cluster size ratio configuration for HDBSCAN.

3.2 Dimensionality Reduction

We reduce the dimensionality of the vectors with two methods in conjunction: (i) An autoencoder, and (ii) the UMAP algorithm (McInnes et al., 2018), a type of manifold learning similar to t-SNE. We choose the same autoencoder setup as previous literature (McConville et al., 2019) with dimensions of the layers $[d, 500, 500, 2000, 20]$ for the encoder, where $d = 768$ in our case, and mirrored dimensions for the decoder; the compressed dimensionality is thus 20. Using UMAP, we reduce this further down to 10. UMAP is a method for general purpose dimensionality reduction that uses local manifold approximations to construct a topological representation of high dimensional data. We decided to use UMAP over t-SNE and PCA as it scales nicely in computation time with the amount of data and produces a better resulting embedding. Additionally, McConville et al. (2019) showed that using an autoencoder before UMAP leads to higher quality clustering and that their approach in combination with a standard clustering approach is competitive with other unsupervised deep learning clustering method and Reif et al. (2019) found that MBERT embeddings projected with UMAP produces distinct clusters of different word senses. Furthermore, there is ample evidence that cosine similarity is the superior metric over Euclidian distance to calculate the similarity of semantic vectors. We incorporate this into our system by using the cosine metric to project our data with UMAP, i.e., preserving the cosine similarity between vectors. Another advantage of using the lower dimensional embeddings is that this allows the use of different clustering methods such as HDBSCAN, for which high input dimensionality is prohibitive.

Our final hyperparameter configuration of UMAP for each language used six neighbours for English and German, five for Swedish and three neighbours for Latin, as shown in Table 4. The number of neighbours dictates how many points the algorithm groups with any other point on the embedding manifold and are thus closer. Other hyperparameters include the minimal distance (0.0) between points in the embedding space, which should only be greater than zero for visualization purposes, and the number of components (which we set to 10). The 10 dimensional vectors are then fed to the subsequent clustering step.

In the post-evaluation, we tested the influence of the dimensionality reduction methods with an ablation study by changing modules in the pipeline, while keeping the same hyperparameters. We compare two settings against the submitted system. First, we omit the autoencoder and only use UMAP in combination with HDBSCAN. Second, we omit dimensionality reduction altogether and only use GMM clustering (n clusters = 5) on the high dimensional data (since HDBSCAN cannot cluster such high dimensional data). We notice that peak performance is barely affected across the setups, but find that additional dimensionality reduction steps improve average performance, while standard deviation becomes more narrow. This indicates that the dimensionality reduction contributes to the stability of the system overall.

3.3 Clustering & Measuring Lexical Semantic Change

We mainly compare two different clustering approaches: (i) Gaussian Mixture Models (GMM) (Reynolds, 2009) and (ii) Hierarchical Density-based clustering (HDBSCAN) (McInnes et al., 2017). We also experimented with a non-hierarchical DBSCAN (Ester et al., 1996) and BIRCH (balanced iterative reducing and clustering using hierarchies) (Zhang et al., 1997). However, non-hierarchical DBSCAN is very sensitive to scale since its hyperparameter epsilon (maximum distance between two points) is a fixed value. BIRCH tended to find too many marginally small clusters that would be treated as noise and had too many hyperparameters overall. In contrast, HDBSCAN, while also hierarchical, has only one hyperparameter.

To retrieve distinct senses for a particular word, all instances for this word in one time epoch are fed as 10 dimensional vectors to different clustering algorithms. An important problem in the given setting is the missing information on the amount of senses. These can be either heuristically set to a fixed number (as for GMM) or inferred via structure learning where a likelihood score for a model on the data given some complexity penalty is optimized (as for HDBSCAN).

As a result of the clustering, each word occurrence is assigned a cluster index, which represents a sense label in our case. For the classification task of whether a sense has been lost/gained or not, we look at all cases where a cluster has at most k contextualized vectors from one time epoch. If that cluster has at least n vectors from the respective other epoch, we say that a sense change has occurred. For subtask 2, we compute, for each word and time epoch, the relative frequency profile of its contextualized embeddings in different clusters. We then compute Jensen-Shannon Distance between the two distributions corresponding to both time epochs and rank words by this metric. An important decision in this model is to handle all points classified as noise as one sense cluster. This decision is based on the assumption that the model could have missed a change if in one epoch there is considerably more noise than in the other.

GMM: GMMs belong to a class of probabilistic models that represent a distribution with many normally distributed subdistributions. GMMs use a fixed amount of clusters, where a cluster is represented by its center in the latent space and covariance structure information. A clustering can then be found by using the Expectation Maximization Algorithm (Moon, 1996). Since GMMs do not search for the number of clusters itself, we manually set the number of clusters.

HDBSCAN: HDBSCAN is a spatial clustering algorithm that is well suited to handle arbitrarily shaped clusters of varying size or density, while also offering mechanisms to automatically identify noise. The only hyperparameter to set is the minimum ratio of points in a cluster—the fraction of the minimal number of points that HDBSCAN needs to construct a cluster. HDBSCAN performs clustering based on the density regions in a dataset and in contrast to the classical DBSCAN algorithm, it is able to find clusters of varying densities, which might be the case for our sense clusters. An additional benefit is that HDBSCAN has an inherent notion of noise data points. This helps deal with 'unnecessary senses' or other outliers.

#cluster	Subtask 1	Subtask 2
3	.547 ± .017	.213 ± .043
5	.536 ± .020	.194 ± .028
10	.514 ± .018	.198 ± .017

Table 5: The average performance on both subtasks of the GMM clustering with different fixed numbers of clusters. The variance is reported from 10 runs of the system.

Cluster size ratio	Subtask 1	Subtask 2
0.01	.501 ± .006	.176 ± .007
0.02	.518 ± .006	.165 ± .010
0.03	.542 ± .020	.210 ± .023
0.04	.535 ± .025	.191 ± .029

Table 6: HDBSCAN: Effect of the minimum cluster size ratio on the performance of the model across different thresholds.

In Tables 5 and 6, we evaluate the impact of hyperparameter choices for both clustering algorithms on the post-evaluation data. For GMM, this is the number of clusters and for HDBSCAN, this is the cluster size ratio. The GMM approach does best with three clusters and shows decreasing performance with an increasing amount of clusters. For HDBSCAN, the optimal cluster size ratio may neither be too small (to avoid having too many senses) nor too large (to not miss senses with few samples). Furthermore, while increasing the required minimum cluster size tends to increase performance for both subtasks, the results show a slightly higher variance, indicating that the model loses robustness with higher values of the cluster size ratio.

4 Conclusion

Our contribution to the SemEval-2020 "Unsupervised Lexical Semantic Change Detection task" identifies lexical semantic sense change in an unsupervised, knowledge-free, and language-agnostic manner. It utilizes contextualized word embeddings to determine implicit sense information from the context of words. These contextual word vectors are reduced in dimensionality with an autoencoder and UMAP to enable

clustering with a hierarchical density based clustering (HDBSCAN), which is an appealing clustering algorithm compared with other, more complex architectures. We found that our chosen hyperparameters for English yielded good results, and further post-evaluation confirmed this. However, for the other languages, our hyperparameter configuration was not optimal, as evidenced by improved performance in post-evaluation with other hyperparameter choices.

Overall, we showed that, for the given task of semantic change analysis, we can achieve best or near-best performance for one of the shared task's languages even when massivly condensing the information from MBERT's contextualized embeddings, yielding a very time- and resource-efficient solution to the problem (that has a memory footprint of 1% of the original embeddings after the dimensionality reduction). That our system has rather mediocre performance for the other languages may also hint at a certain randomness in the task definition (e.g., w.r.t. the parameters k and n) or as arising from the rather small sizes or selection of some of the involved corpora or their annotation process.

References

Andy Coenen, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda Viégas, and Martin Wattenberg. 2019. Visualizing and measuring the geometry of bert. *arXiv preprint arXiv:1906.02715*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Steffen Eger and Alexander Mehler. 2016. On the linearity of semantic change: Investigating meaning variation via dynamic graph models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 52–58, Berlin, Germany, August. Association for Computational Linguistics.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

Christiane Fellbaum. 2012. Wordnet. *The encyclopedia of applied linguistics*.

Mario Giulianelli. 2019. Lexical semantic change analysis with contextualised word representations. *Unpublished master's thesis, University of Amsterdam, Amsterdam*.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016a. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2016, page 2116. NIH Public Access.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*.

Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic sense modeling with deep contextualized word embeddings: An ecological view. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3899–3908.

Ryan McConville, Raul Santos-Rodriguez, Robert J Piechocki, and Ian Craddock. 2019. N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding. *arXiv preprint arXiv:1908.05968*.

Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *The Journal of Open Source Software*, 2(11):205.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Todd K Moon. 1996. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B Viegas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of bert. In *Advances in Neural Information Processing Systems*, pages 8592–8600.

Douglas A Reynolds. 2009. Gaussian mixture models. *Encyclopedia of biometrics*, 741.

Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. 2018. Diachronic usage relatedness (durel): A framework for the annotation of lexical semantic change. *arXiv preprint arXiv:1804.06517*.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Tian Zhang, Raghu Ramakrishnan, and Miron Livny. 1997. Birch: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182.