Definition generation for lexical semantic change detection

Mariia Fedorova, Andrey Kutuzov, Yves Scherrer
Language Technology Group
University of Oslo

ACL Findings, 2024

Definition generation for semantic change detection

Definitions as senses

- We use LLM-generated contextualized word definitions for lexical semantic change detection (LSCD)
  - ...by comparing vectorized definitions
  - ...by comparing the distributions of definitions ('definitions-as-senses')
- Dictionary-like definitions give us interpretability:
  - what exactly has changed in the meaning of a word over time?

Definition generation for semantic change detection

Definitions as senses

- We use LLM-generated contextualized word definitions for lexical semantic change detection (LSCD)
  - ...by comparing vectorized definitions
  - ...by comparing the distributions of definitions ('definitions-as-senses')
- Dictionary-like definitions give us interpretability:
  - what exactly has changed in the meaning of a word over time?
- Still reasonable performance on three different languages.

Definition generation for semantic change detection

Definitions as senses

- We use LLM-generated contextualized word definitions for lexical semantic change detection (LSCD)
  - ...by comparing vectorized definitions
  - ...by comparing the distributions of definitions ('definitions-as-senses')
- Dictionary-like definitions give us interpretability:
  - what exactly has changed in the meaning of a word over time?
- Still reasonable performance on three different languages.

Important prior work

- the general idea of using definitions as representations from [Giulianelli et al., 2023]

Definition generation for semantic change detection

Definitions as senses

- We use LLM-generated contextualized word definitions for lexical semantic change detection (LSCD)
  - ...by comparing vectorized definitions
  - ...by comparing the distributions of definitions ('definitions-as-senses')
- Dictionary-like definitions give us interpretability:
  - what exactly has changed in the meaning of a word over time?
- Still reasonable performance on three different languages.

Important prior work

- the general idea of using definitions as representations from [Giulianelli et al., 2023]
- baseline sense-based LSCD system from [Tang et al., 2023]
  - (they used pre-defined sense inventories, no definition generation)

Definition generation for semantic change detection

Definitions as senses

- We use LLM-generated contextualized word definitions for lexical semantic change detection (LSCD)
  - ...by comparing vectorized definitions
  - ...by comparing the distributions of definitions ('definitions-as-senses')
- Dictionary-like definitions give us interpretability:
  - what exactly has changed in the meaning of a word over time?
- Still reasonable performance on three different languages.

Important prior work

- the general idea of using definitions as representations from [Giulianelli et al., 2023]
- baseline sense-based LSCD system from [Tang et al., 2023]
  - (they used pre-defined sense inventories, no definition generation)
- fine-tuned definition generation models from [Kutuzov et al., 2024]

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

2 sets of generated definitions (for 2 time periods). How to compare them for a change score?

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

2 sets of generated definitions (for 2 time periods). How to compare them for a change score?

Method 1. Definition embeddings

1. Embed the generated definitions into a vector space with, e.g., `DistilRoBERTa`

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

2 sets of generated definitions (for 2 time periods). How to compare them for a change score?

Method 1. Definition embeddings

1. Embed the generated definitions into a vector space with, e.g., `DistilRoBERTa`
2. Use well-developed LSCD methods of measuring distances between representations: PRT, APD, PRT/APD [Kutuzov et al., 2022]

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

2 sets of generated definitions (for 2 time periods). How to compare them for a change score?

Method 1. Definition embeddings

1. Embed the generated definitions into a vector space with, e.g., `DistilRoBERTa`
2. Use well-developed LSCD methods of measuring distances between representations: PRT, APD, PRT/APD [Kutuzov et al., 2022]
3. The only difference: definition embeddings instead of token embeddings

Our systems

General pipeline

1. The task: to rank a given set of words by the degree of their semantic change in two time periods (represented by two corpora)
2. Let's find the senses of all the target word occurrences!
   - [Tang et al., 2023] did this by choosing a sense from an ontology (WSD)
   - we instead generate a definition from scratch for each target word occurrence...
   - ...assuming that different definitions will be generated for different senses.

2 sets of generated definitions (for 2 time periods). How to compare them for a change score?

Method 1. Definition embeddings

1. Embed the generated definitions into a vector space with, e.g., `DistilRoBERTa`
2. Use well-developed LSCD methods of measuring distances between representations: PRT, APD, PRT/APD [Kutuzov et al., 2022]
3. The only difference: definition embeddings instead of token embeddings

Good performance, but interpretability is lost after vectorizing definitions.

Method 2. Definitions as strings

- Let's use definitions in their textual form
- Every definition is a 'sense'
- definition distributions for 2 time periods are compared similarly to how sense distributions are compared in [Tang et al., 2023]

Method 2. Definitions as strings

- Let's use definitions in their textual form
- Every definition is a 'sense'
- definition distributions for 2 time periods are compared similarly to how sense distributions are compared in [Tang et al., 2023]
- These 'senses' are too granular, because of too many unique definitions. E.g., 'plane':
  - '*An aircraft, especially one designed for military use*'
  - '*An aircraft, especially a military aircraft*', etc

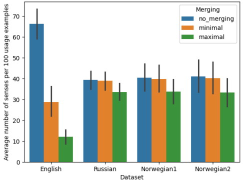
Method 2. Definitions as strings

- Let's use definitions in their textual form
- Every definition is a 'sense'
- definition distributions for 2 time periods are compared similarly to how sense distributions are compared in [Tang et al., 2023]
- These 'senses' are too granular, because of too many unique definitions. E.g., 'plane':
  - '*An aircraft, especially one designed for military use*'
  - '*An aircraft, especially a military aircraft*', etc
- Hence, we merge similar definitions into one:
  - 'Minimalist merging': only with the most frequent definition
  - 'Full-fledged merging': any pair of definitions can be merged together
- Merging if Levenshtein edit distance exceeds a pre-defined threshold.

Method 2. Definitions as strings

- Let's use definitions in their textual form
- Every definition is a 'sense'
- definition distributions for 2 time periods are compared similarly to how sense distributions are compared in [Tang et al., 2023]
- These 'senses' are too granular, because of too many unique definitions. E.g., 'plane':
  - '*An aircraft, especially one designed for military use*'
  - '*An aircraft, especially a military aircraft*', etc
- Hence, we merge similar definitions into one:
  - 'Minimalist merging': only with the most frequent definition
  - 'Full-fledged merging': any pair of definitions can be merged together
- Merging if Levenshtein edit distance exceeds a pre-defined threshold.

Merging reduces the number of unique definitions: more realistic as senses.



Average number of senses per 100 usages before and after merging, calculated across all datasets for each language.
(should be done cautiously: merging too much can degrade the performance)

One English example

3 most frequent definitions/senses per time period for *ball* (high predicted change rate):

	Period 1 (1810-1860)	**Period 2** (1960-2010)
ball Jensen-Shannon divergence (change score): 0.83	*A spherical object especially one that is round in shape* (82%)	*The object hit in a game* (80%)
	A party (6%)	*The object used in various sports especially in soccer tennis basketball etc* (<1%)
	A wedding (<1%)	*The object used in various sports especially in soccer basketball and other games which is thrown or kicked* (<1%)

One English example

3 most frequent definitions/senses per time period for *ball* (high predicted change rate):

	Period 1 (1810-1860)	**Period 2** (1960-2010)
ball Jensen-Shannon divergence (change score): 0.83	*A spherical object especially one that is round in shape* (82%)	*The object hit in a game* (80%)
	A party (6%)	*The object used in various sports especially in soccer tennis basketball etc* (<1%)
	A wedding (<1%)	*The object used in various sports especially in soccer basketball and other games which is thrown or kicked* (<1%)

More examples, data and code at
`https://github.com/ltgoslo/Definition-generation-for-LSCD`

Performance across different generation and merging strategies

	English		**Norwegian-1**		**Norwegian-2**		**Russian-1**		**Russian-2**		**Russian-3**	
	Cosine	JS	Cosine	JS	Cosine	JS	Cosine	JS	Cosine	JS	Cosine	JS
No merging:												
Greedy	0.461	0.405	*0.303*	0.332	*0.211*	*0.232*	0.299	0.390	0.337	0.427	0.383	0.469
Beam	0.457	0.476	*0.268*	*0.238*	*0.216*	*0.201*	0.304	0.368	0.297	0.403	0.317	0.417
Diverse	0.449	0.382	*0.241*	*0.280*	*0.069*	*0.164*	0.301	0.345	0.310	0.389	0.348	0.421
Minimalist merging:												
Greedy	0.564	**0.565**	*0.251*	*0.280*	*0.192*	*0.197*	0.271	**0.391**	0.233	**0.431**	0.325	**0.491**
Beam	0.510	0.500	*0.297*	*0.240*	*0.112*	*0.189*	0.298	0.366	0.252	0.383	0.301	0.409
Diverse	0.478	0.434	0.325	*0.296*	*0.162*	*0.215*	0.265	0.354	0.268	0.406	0.287	0.443
Full-fledged merging:												
Greedy	0.417	0.418	*0.261*	**0.362**	*0.193*	***0.260***	0.286	**0.391**	0.250	0.416	0.360	0.476
Beam	0.492	0.493	*0.265*	*0.215*	*0.186*	*0.226*	0.304	0.360	0.250	0.347	0.327	0.420
Diverse	*0.312*	*0.301*	*0.209*	*0.315*	*0.202*	*0.221*	0.236	0.301	0.217	0.379	0.262	0.411
Threshold	50		10		10		10		10		10	

LSCD performance (Spearman's ρ) with different generation, merging and distance calculation strategies.

Threshold: Levenshtein edit distance threshold for merging definitions.

Results (Spearman's ρ for graded LSCD) compared to the baselines

Method	**English**	**Norwegian-1**	**Norwegian-2**	**Russian-1**	**Russian-2**	**Russian-3**
			Non-interpretable methods:			
XLM-R token embeddings	0.514$^{\diamondsuit}$	0.394$^{\diamondsuit}$	0.387$^{\diamondsuit}$	0.376$^{\diamondsuit}$	**0.480**$^{\diamondsuit}$	0.457$^{\diamondsuit}$
XL-LEXEME+APD (WiC-based)	0.886	0.659	0.640	0.796	0.820	0.863
Definition embeddings (ours)	**0.637**	**0.496**	**0.565**	**0.488**	0.462	**0.504**
			Interpretable methods:			
Lesk without PoS	0.423$^{\clubsuit}$	*0.178*	0.500	0.294	0.279	0.286
Lesk with PoS	0.587	*0.150*	0.474			
ARES sense embeddings	0.529$^{\clubsuit}$	—	—	—	—	—
LMMS sense embeddings	0.589$^{\clubsuit}$	—	—	—	—	—
Definitions as senses (ours)	0.565	0.362	*0.260*	0.391	0.431	0.491

($\diamondsuit$: best results from [Giulianelli et al., 2022], $\clubsuit$: [Tang et al., 2023], XL-LEXEME results from [Periti and Tahmasebi, 2024])

Definition embeddings yield better performance, but definitions as senses are interpretable and explainable.

References I

Giulianelli, M., Kutuzov, A., and Pivovarova, L. (2022).
Do not fire the linguist: Grammatical profiles help language models detect semantic change.
In Tahmasebi, N., Montariol, S., Kutuzov, A., Hengchen, S., Dubossarsky, H., and Borin, L., editors, Proceedings of the 3rd Workshop on Computational Approaches to Historical Language Change, pages 54–67, Dublin, Ireland. Association for Computational Linguistics.

Giulianelli, M., Luden, I., Fernandez, R., and Kutuzov, A. (2023).
Interpretable word sense representations via definition generation: The case of semantic change analysis.
In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3130–3148, Toronto, Canada. Association for Computational Linguistics.

References II

Kutuzov, A., Fedorova, M., Schlechtweg, D., and Arefyev, N. (2024).
Enriching word usage graphs with cluster definitions.
In Calzolari, N., Kan, M.-Y., Hoste, V., Lenci, A., Sakti, S., and Xue, N., editors, Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 6189–6198, Torino, Italia. ELRA and ICCL.

Kutuzov, A., Velldal, E., and Øvrelid, L. (2022).
Contextualized embeddings for semantic change detection: Lessons learned.
In Derczynski, L., editor, Northern European Journal of Language Technology, Volume 8, Copenhagen, Denmark. Northern European Association of Language Technology.

References III

Periti, F. and Tahmasebi, N. (2024).
A systematic comparison of contextualized word embeddings for lexical semantic change.
In Duh, K., Gomez, H., and Bethard, S., editors, Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 4262–4282, Mexico City, Mexico. Association for Computational Linguistics.

Tang, X., Zhou, Y., Aida, T., Sen, P., and Bollegala, D. (2023).
Can word sense distribution detect semantic changes of words?
In Bouamor, H., Pino, J., and Bali, K., editors, Findings of the Association for Computational Linguistics: EMNLP 2023, pages 3575–3590, Singapore. Association for Computational Linguistics.