Computational approaches to semantic change detection
Day 2
Learned representations for semantic change detection
part 1: static embeddings

Andrey Kutuzov, Lidia Pivovarova

University of Oslo, University of Helsinki
ESSLLI’2023

Contents

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]
- Can also analyze synchronic cross-domain semantic shifts [Kutuzov and Kuzmenko, 2015].

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]
- Can also analyze synchronic cross-domain semantic shifts [Kutuzov and Kuzmenko, 2015].

How to automate this?

- Distributional hypothesis [Firth, 1957]:
- Word meaning $\approx$ word contexts

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]
- Can also analyze synchronic cross-domain semantic shifts [Kutuzov and Kuzmenko, 2015].

How to automate this?

- Distributional hypothesis [Firth, 1957]:
- Word meaning ≈ word contexts
- Changes in contexts ≈ changes in meaning

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]
- Can also analyze synchronic cross-domain semantic shifts [Kutuzov and Kuzmenko, 2015].

How to automate this?

- Distributional hypothesis [Firth, 1957]:
- Word meaning $\approx$ word contexts
- Changes in contexts $\approx$ changes in meaning
- Temporal cultural and linguistic changes influence the contexts

Introduction

Diachronic semantic change

- Words change their meaning over time:
  - a.k.a. diachronic semantic shifts.
- Our main task:
  - Diachronic semantic shift detection
  - Lexical semantic change detection (LSCD) [Schlechtweg et al., 2019]
- Can also analyze synchronic cross-domain semantic shifts [Kutuzov and Kuzmenko, 2015].

How to automate this?

- Distributional hypothesis [Firth, 1957]:
- Word meaning $\approx$ word contexts
- Changes in contexts $\approx$ changes in meaning
- Temporal cultural and linguistic changes influence the contexts
- Let's use distributional word embeddings to trace these changes!

Contents

Word embeddings per se

Word embeddings

- Distributional semantic representations efficiently capture word meaning
  - static word embeddings or type-based word embeddings: *word2vec* [Mikolov et al., 2013], *fastText* [Bojanowski et al., 2017]
  - every word type is assigned a dense vector, so that semantically similar words have similar vectors (embeddings)
- These representations are trained on large text corpora, usually on the task of language modeling.
- For example, `word2vec CBOW`, context window = 5, vector size 300
- Extremely important for modern NLP.

Word embeddings per se

Word embeddings

- Distributional semantic representations efficiently capture word meaning
  - static word embeddings or type-based word embeddings: *word2vec* [Mikolov et al., 2013], *fastText* [Bojanowski et al., 2017]
  - every word type is assigned a dense vector, so that semantically similar words have similar vectors (embeddings)
- These representations are trained on large text corpora, usually on the task of language modeling.
- For example, `word2vec CBOW`, context window = 5, vector size 300
- Extremely important for modern NLP.

Can be used to capture semantic change in an unsupervised manner.

Contents

Word embeddings in LSCD

- Linguistics: hand-picking examples
  [Traugott and Dasher, 2001, Daniel and Dobrushina, 2016]

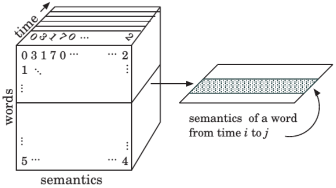
Word embeddings in LSCD

- Linguistics: hand-picking examples

  [Traugott and Dasher, 2001, Daniel and Dobrushina, 2016]
- NLP: large-scale diachronic shift mining using distributional semantic models.

Word embeddings in LSCD

- Linguistics: hand-picking examples
  [Traugott and Dasher, 2001, Daniel and Dobrushina, 2016]
- NLP: large-scale diachronic shift mining using distributional semantic models.
- Word embedding changes largely correspond to diachronic semantic change.
- *'Semantic change equals to cosine distance between the word vectors in time period* 1 *and time period* 2*'*



Tensor representation of a semantic space

[Jurgens and Stevens, 2009]

Word embeddings in LSCD

- NLP practitioners faced several challenges when applying this approach
- Technical:
  - how to make diachronic embedding spaces comparable?
  - how to control for word frequencies?

Word embeddings in LSCD

- NLP practitioners faced several challenges when applying this approach
- Technical:
  - how to make diachronic embedding spaces comparable?
  - how to control for word frequencies?
- Conceptual:
  - different senses and grammatical meanings are expressed in identical word usages
  - ...and are mixed into a word embedding together.

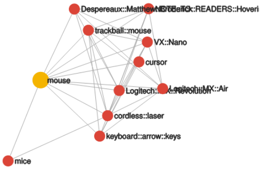
What words are related to **«mouse»** in model «Google News» ?

Word frequency

☑ High ☑ Medium ☐ Low

1. Logitech MX Revolution NOUN 0.6175
2. mice NOUN 0.5897
3. cordless laser NOUN 0.5652
4. VX Nano NOUN 0.5619
5. keyboard arrow keys NOUN 0.5546
6. NOTE TO READERS Hovering NOUN 0.5520
7. cursor NOUN 0.5472
8. Logitech MX Air NOUN 0.5459
9. trackball mouse NOUN 0.5446
10. Despereaux Matthew Broderick NOUN 0.5388

- *We show only the associates of the same part of speech as your query. All associates can be found at the Similar Words tab.*



0.5 Similarity threshold ☐ Show tags

- Still, significant results are achieved.

Word embeddings in LSCD

SemEval-2020 featured the first shared task on LSCD [Schlechtweg et al., 2020]

Task 1: Unsupervised Lexical Semantic Change Detection

- Two subtasks:
  1. classification task
  2. ranking task
- 4 languages: German, English, Swedish, Latin

Word embeddings in LSCD

SemEval-2020 featured the first shared task on LSCD [Schlechtweg et al., 2020]

Task 1: Unsupervised Lexical Semantic Change Detection

- Two subtasks:
  1. classification task
  2. ranking task
- 4 languages: German, English, Swedish, Latin

18 of 21 participants (including all the winners) used word embeddings.

Word embeddings in LSCD

SemEval-2020 featured the first shared task on LSCD [Schlechtweg et al., 2020]

Task 1: Unsupervised Lexical Semantic Change Detection

- Two subtasks:
  1. classification task
  2. ranking task
- 4 languages: German, English, Swedish, Latin

18 of 21 participants (including all the winners) used word embeddings.
Systems employing type-based word embeddings clearly outperformed the rest (token-based word embeddings and topic models).

Contents

Comparing words across static embedding models

Methods of using type-based word embeddings in LSCD are surveyed in [Kutuzov et al., 2018] and [Tang, 2018].

- Problem: embeddings of one and the same word trained in different runs are not comparable
  - especially if trained on different corpora

Comparing words across static embedding models

Methods of using type-based word embeddings in LSCD are surveyed in [Kutuzov et al., 2018] and [Tang, 2018].

- Problem: embeddings of one and the same word trained in different runs are not comparable
  - especially if trained on different corpora
- So, how to find the meaningful difference between $cell_{1910}$ and $cell_{2010}$?
- Various algorithms of making word embeddings actually diachronic:

Comparing words across static embedding models

Methods of using type-based word embeddings in LSCD are surveyed in [Kutuzov et al., 2018] and [Tang, 2018].

- Problem: embeddings of one and the same word trained in different runs are not comparable
  - especially if trained on different corpora
- So, how to find the meaningful difference between $cell_{1910}$ and $cell_{2010}$?
- Various algorithms of making word embeddings actually diachronic:
  - Training models incrementally [Kim et al., 2014]

Comparing words across static embedding models

Methods of using type-based word embeddings in LSCD are surveyed in [Kutuzov et al., 2018] and [Tang, 2018].

- Problem: embeddings of one and the same word trained in different runs are not comparable
  - especially if trained on different corpora
- So, how to find the meaningful difference between $cell_{1910}$ and $cell_{2010}$?
- Various algorithms of making word embeddings actually diachronic:
  - Training models incrementally [Kim et al., 2014]
  - Training separate models for each time period (time bin):
    - Aligning embedding spaces [Hamilton et al., 2016b]
    - Comparing distances between a given word and all others (second-rank similarity) [Yin et al., 2018]
    - Compare only sets of *n* nearest neighbors

Comparing words across static embedding models

Methods of using type-based word embeddings in LSCD are surveyed in [Kutuzov et al., 2018] and [Tang, 2018].

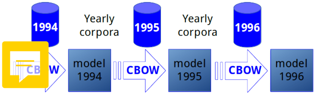
- Problem: embeddings of one and the same word trained in different runs are not comparable
  - especially if trained on different corpora
- So, how to find the meaningful difference between $cell_{1910}$ and $cell_{2010}$?
- Various algorithms of making word embeddings actually diachronic:
  - Training models incrementally [Kim et al., 2014]
  - Training separate models for each time period (time bin):
    - Aligning embedding spaces [Hamilton et al., 2016b]
    - Comparing distances between a given word and all others (second-rank similarity) [Yin et al., 2018]
    - Compare only sets of *n* nearest neighbors
  - Training models jointly across time bins (did not attract much attention) [Bamler and Mandt, 2017, Yao et al., 2018, Rosenfeld and Erk, 2018]
  - ...

Comparing words across static embedding models

Different ways of training ‘diachronic’ word embeddings

For n time bins:

- ‘Incremental’ models
  - Model trained on time bin tb_0,
  - Model trained on time bin tb_1, initialized with tb_0 weights,
  - ...
  - Model trained on time bin tb_n, initialized with tb_{n-1} weights.
- ‘Separate’ models:
  - Model trained on time bin tb_0,
  - Model trained on time bin tb_1,
  - ...
  - Model trained on time bin tb_n



Comparing words across static embedding models

Methods for actually comparing word embeddings fall into two conceptual classes:

Local methods for semantic shift detection

Comparing words' nearest neighbors:

- Jaccard similarity [Jaccard, 1901]
- Kendall's τ [Kendall, 1948]

Global methods for semantic shift detection

Comparing words' vectors (or semantic spaces in general):

- Procrustes alignment [Hamilton et al., 2016b]
- Global Anchors [Yin et al., 2018]

Comparing words across static embedding models

Methods for actually comparing word embeddings fall into two conceptual classes:

Local methods for semantic shift detection

Comparing words' nearest neighbors:

- Jaccard similarity [Jaccard, 1901]
- Kendall's τ [Kendall, 1948]

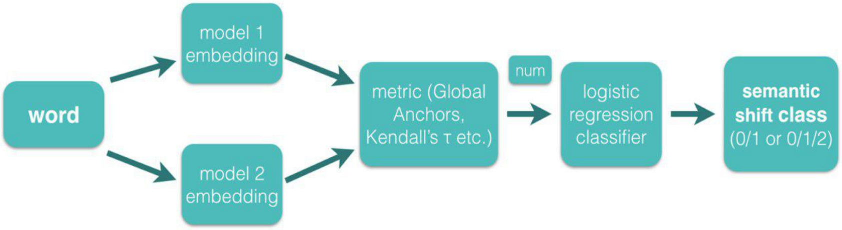
Global methods for semantic shift detection

Comparing words' vectors (or semantic spaces in general):

- Procrustes alignment [Hamilton et al., 2016b]
- Global Anchors [Yin et al., 2018]

...and many more!

Comparing words across static embedding models



General workflow for an LSCD method using type-based word embeddings.

Local methods

Jaccard similarity

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} \tag{1}$$

[Jaccard, 1901]

Nearest neighbors for ‘*quarantine*’:

- X = ‘*prison, illness, plague, ship*’
- Y = ‘*self-isolation, covid, regime, epidemics*’

$J(X, Y) = 0$

Can you guess the years for X and Y?

Local methods

Kendall's τ

Takes into account the ranking of neighbors [Kendall, 1948]

$$\tau = \frac{2}{n(n-1)} \sum_{i<j} sgn(x_i - x_j)\, sgn(y_i - y_j) \tag{2}$$

Nearest neighbors for '*corona*' ($x = 2019, y = 2021$):

$$\begin{aligned} x_1 = crown &\Rightarrow y_1 = virus \\ \ldots\ldots &\Rightarrow \ldots\ldots \\ x_{10} = virus &\Rightarrow y_{10} = crown \end{aligned} \tag{3}$$

Local methods

Kendall's τ

Takes into account the ranking of neighbors [Kendall, 1948]

$$\tau = \frac{2}{n(n-1)} \sum_{i<j} sgn(x_i - x_j)\, sgn(y_i - y_j) \tag{2}$$

Nearest neighbors for '*corona*' ($x = 2019, y = 2021$):

$$\begin{aligned} x_1 &= crown \Rightarrow y_1 = virus \\ &\ldots\ldots \Rightarrow \ldots\ldots \\ x_{10} &= virus \Rightarrow y_{10} = crown \end{aligned} \tag{3}$$

Local Neighborhood Distance: similarity between target word similarities to n nearest neighbors [Hamilton et al., 2016a].

Global methods

Orthogonal Procrustes Analysis

- First, we ‘align’ two models [Hamilton et al., 2016b].
- Given embedding matrices A and B, find an orthogonal matrix R that maps A to B via singular value decomposition, SVD.
- Then, simple cosine similarity between $word^A$ and $word^B$ is calculated.

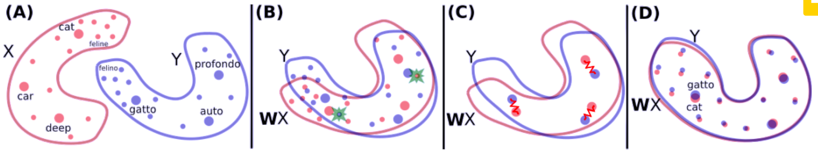
$$R = \underset{R}{\operatorname{argmin}} ||R \cdot A - B||^2 \tag{4}$$

Global methods

Orthogonal Procrustes Analysis

- First, we 'align' two models [Hamilton et al., 2016b].
- Given embedding matrices A and B, find an orthogonal matrix R that maps A to B via singular value decomposition, SVD.
- Then, simple cosine similarity between $word^A$ and $word^B$ is calculated.

$$R = \underset{R}{\operatorname{argmin}} ||R \cdot A - B||^2 \qquad (4)$$



Aligning English and Italian word embeddings for machine translation

Global methods

Global Anchors

[Yin et al., 2018] define semantic shift of word *w* from year *x* to year *y* like this:

$$\begin{aligned} similarities_x &= (x_1, ..., x_n) \\ similarities_y &= (y_1, ..., y_n) \end{aligned} \tag{5}$$

- x_i and y_i are cosine similarities between the word *w* and the i$^{\text{th}}$ word in the intersection of *x* and *y* vocabularies (of size *n*).

Global methods

Global Anchors

[Yin et al., 2018] define semantic shift of word w from year x to year y like this:

$$\begin{aligned} similarities_x &= (x_1, ..., x_n) \\ similarities_y &= (y_1, ..., y_n) \end{aligned} \tag{5}$$

- x_i and y_i are cosine similarities between the word w and the ith word in the intersection of x and y vocabularies (of size n).
- We compare global positions of w in the semantic space.
- Semantic similarity between different time periods = $cos(similarities_x, similarities_y)$
- With Global Anchors, no explicit alignment needed.

Comparing words across static embedding models

Temporal referencing

- Time labels as tags [Dubossarsky et al., 2019]
- When training a `word2vec` model, each target word is replaced with a time-specific token:
  - in the 1920s corpus: *computer* $\rightarrow$ $\textit{computer}_{1920}$

Comparing words across static embedding models

Temporal referencing

- Time labels as tags [Dubossarsky et al., 2019]
- When training a `word2vec` model, each target word is replaced with a time-specific token:
  - in the 1920s corpus: *computer* $\rightarrow$ $computer_{1920}$
- If it is a context word, it remains unchanged.

Comparing words across static embedding models

Temporal referencing

- Time labels as tags [Dubossarsky et al., 2019]
- When training a `word2vec` model, each target word is replaced with a time-specific token:
  - in the 1920s corpus: *computer* $\rightarrow$ *computer*$_{1920}$
- If it is a context word, it remains unchanged.
- One vector space is learned for all time periods.
- Change is measured as cosine distance between *computer*$_{1920}$ and *computer*$_{1980}$ vectors.
- Again, no post-hoc alignment necessary!

Comparing words across static embedding models

Temporal referencing

- Time labels as tags [Dubossarsky et al., 2019]
- When training a `word2vec` model, each target word is replaced with a time-specific token:
  - in the 1920s corpus: *computer* $\rightarrow$ *computer*$_{1920}$
- If it is a context word, it remains unchanged.
- One vector space is learned for all time periods.
- Change is measured as cosine distance between *computer*$_{1920}$ and *computer*$_{1980}$ vectors.
- Again, no post-hoc alignment necessary!

Contradicting reports: [Schlechtweg et al., 2019] say it fails on German data.

Contents

Empirical results on static word embeddings

- Global methods work better for semantic change detection
- Procrustes alignment is clearly the best [Schlechtweg et al., 2019, Schlechtweg et al., 2020]
  - clearly wins on granular time spans
  - criticised for instability with respect to different embedding spaces [Gonen et al., 2020].
- Local methods are still applicable (but sometimes worse than random)
- Combining methods is a good idea
- Incrementally trained models are worse than separate models aligned to the common space
  - incremental training is largely abandoned now.

Empirical results on static word embeddings

- Global methods work better for semantic change detection
- Procrustes alignment is clearly the best [Schlechtweg et al., 2019, Schlechtweg et al., 2020]
  - clearly wins on granular time spans
  - criticised for instability with respect to different embedding spaces [Gonen et al., 2020].
- Local methods are still applicable (but sometimes worse than random)
- Combining methods is a good idea
- Incrementally trained models are worse than separate models aligned to the common space
  - incremental training is largely abandoned now.
- Some issues are addressed by using contextualized embeddings (pre-trained large language models). More on this tomorrow!

Practical session

Thanks! Any questions?

And then we will get our hands dirty with word embeddings for LSCD.

https://github.com/lmphcs/semshift_esslli2023

References I

Bamler, R. and Mandt, S. (2017).
Dynamic word embeddings.
In *Proceedings of the International Conference on Machine Learning*, pages 380–389, Sydney, Australia.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017).
Enriching word vectors with subword information.
Transactions of the Association for Computational Linguistics, 5:135–146.

Daniel, M. and Dobrushina, N. (2016).
Two centuries in twenty words (in Russian).
NRU HSE.

References II

- Dubossarsky, H., Hengchen, S., Tahmasebi, N., and Schlechtweg, D. (2019).
  Time-out: Temporal referencing for robust modeling of lexical semantic change.
  In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 457–470, Florence, Italy. Association for Computational Linguistics.

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

- Gonen, H., Jawahar, G., Seddah, D., and Goldberg, Y. (2020).
  Simple, interpretable and stable method for detecting words with usage change across corpora.
  In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 538–555, Online. Association for Computational Linguistics.

References III

- Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016a).
  Cultural shift or linguistic drift? comparing two computational measures of semantic change.
  In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2116–2121, Austin, Texas. Association for Computational Linguistics.

- Hamilton, W. L., Leskovec, J., and Jurafsky, D. (2016b).
  Diachronic word embeddings reveal statistical laws of semantic change.
  In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1501, Berlin, Germany. Association for Computational Linguistics.

References IV

Jaccard, P. (1901).
Distribution de la Flore Alpine: dans le Bassin des dranses et dans quelques régions voisines.
Rouge.

Jurgens, D. and Stevens, K. (2009).
Event detection in blogs using temporal random indexing.
In *Proceedings of the Workshop on Events in Emerging Text Types*, pages 9–16, Borovets, Bulgaria. Association for Computational Linguistics.

Kendall, M. G. (1948).
Rank correlation methods.
Griffin.

References V

- Kim, Y., Chiu, Y.-I., Hanaki, K., Hegde, D., and Petrov, S. (2014).
  Temporal analysis of language through neural language models.
  In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 61–65, Baltimore, MD, USA. Association for Computational Linguistics.

- Kutuzov, A. and Kuzmenko, E. (2015).
  Comparing neural lexical models of a classic national corpus and a web corpus: The case for Russian.
  Lecture Notes in Computer Science, 9041:47–58.

References VI

Kutuzov, A., Øvrelid, L., Szymanski, T., and Velldal, E. (2018).
Diachronic word embeddings and semantic shifts: a survey.
In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013).
Distributed representations of words and phrases and their compositionality.
Advances in Neural Information Processing Systems, 26:3111–3119.

References VII

Rosenfeld, A. and Erk, K. (2018).
Deep neural models of semantic shift.
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 474–484, New Orleans, Louisiana. Association for Computational Linguistics.

Schlechtweg, D., Hätty, A., Del Tredici, M., and Schulte im Walde, S. (2019).
A wind of change: Detecting and evaluating lexical semantic change across times and domains.
In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 732–746, Florence, Italy. Association for Computational Linguistics.

References VIII

Schlechtweg, D., McGillivray, B., Hengchen, S., Dubossarsky, H., and Tahmasebi, N. (2020).
SemEval-2020 task 1: Unsupervised lexical semantic change detection.
In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1–23, Barcelona (online). International Committee for Computational Linguistics.

Tang, X. (2018).
A state-of-the-art of semantic change computation.
Natural Language Engineering, 24(5):649–676.

Traugott, E. C. and Dasher, R. B. (2001).
Regularity in semantic change.
Cambridge University Press.

References IX

Yao, Z., Sun, Y., Ding, W., Rao, N., and Xiong, H. (2018).
Dynamic word embeddings for evolving semantic discovery.
In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 673–681, Marina Del Rey, CA, USA.

Yin, Z., Sachidananda, V., and Prabhakar, B. (2018).
The global anchor method for quantifying linguistic shifts and domain adaptation.
In *Advances in Neural Information Processing Systems*, pages 9433–9444.