UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOCRITOS"
MSC PROGRAMME IN DATA SCIENCE

The impact of different representations in the presence of language drift

by

Ioannis Christodoulou

A thesis submitted in partial fulfillment
of the requirements for the MSc
in Data Science

Supervisor: Ilias Zavitsanos
Post-Doctoral Researcher

Co-supervisors: George Giannakopoulos,
Research Fellow,

Athens, July 2022

The impact of different representations in the presence of language drift

Ioannis Christodoulou

MSc. Thesis, MSc. Programme in Data Science

University of the Peloponnese & NCSR “Democritos”, July 2022

UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOCRITOS"
MSC PROGRAMME IN DATA SCIENCE

The impact of different representations in the presence of language drift

by

Ioannis Christodoulou

A thesis submitted in partial fulfillment
of the requirements for the MSc
in Data Science

Supervisor: Ilias Zavitsanos
Post-Doctoral Researcher

Co-supervisors: George Giannakopoulos,
Research Fellow,

Approved by the examination committee on July, 2022.

(Signature)	(Signature)	(Signature)
.....	.....	.....
Ilias Zavitsanos Post-Doctoral Researcher	George Giannakopoulos Research Fellow	Anastatis Krithara Post-Doctoral Researcher

Athens, July 2022

UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOCRITOS"
MSC PROGRAMME IN DATA SCIENCE

Declaration of Authorship

1. I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

2. I confirm that this thesis presented for the degree of Master of Science in Data Science, has

   (a) been composed entirely by myself

   (b) been solely the result of my own work

   (c) not been submitted for any other degree or professional qualification

3. I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or processional qualification except as specified.

(Signature)

.....

Ioannis Christodoulou

Athens, July 2022

Acknowledgments

Foremost, I would like to thank my supervisor of the Master's thesis, Post-Doctoral Researcher Ilias Zavitsanos, for his guidance, continuous support, and insightful comments throughout the research and writing of this thesis. Besides my supervisor, I would like to particularly thank my co-supervisor, Research Fellow George Giannakopoulos, who was always there to listen and advise me on the research methodology. I am also thankful to the REPR research group, where I had the opportunity to participate, exchange ideas and learn from the works of the other members. Last but not least, I would like to thank my family and friends for their support and help. Without them, this thesis would not have been possible.

Abstract

Natural language inherently contains an interpretation of the world in the form of vocabulary and the different meanings of words. Language changes can reflect sociocultural evolution; therefore, their systematical exploration is a valuable tool to social and humanities sciences researchers. In this thesis, we examine the detection of semantic changes between two time periods *t1*, *t2*. For the empirical study, we use datasets of four different languages (English, German, Latin, and Swedish) provided from the SemEval-2020 Task 1. The whole set of our experiments is evaluated against a binary classification task, depending on whether a word's sense changes or not. For that purpose, we explore a set of different approaches including methods that have not been previously submitted in the SemEval-2020 Task 1. Furthermore, we create an extensible system which decouples each stage of the diachronic semantic change detection workflow from the actual implementations. This approach contributes to a quick and efficient reproduction of the experiments, aiming to facilitate research in the domain of semantic change. Based on the results of our empirical study, we answer three different questions. The first is related to identifying the most suitable alignment method for the word embeddings *Wt1*, *Wt2*. The methods under investigation are the *Orthogonal Procrustes*, the *Incremental Training*, and the *Temporal Word Embeddings with a Compass*. The next question refers to the performance of the *Word2vec* pre-trained embeddings compared to others whose weights had not been prior initialized. Finally, through the application of *LDA2vec*, we explore whether the *LDA (Latent Dirichlet Allocation)* topics improve the performance of the *SGNS (Skip-gram with Negative Sampling)* or not.

Contents

List of Tables iii

List of Figures iv

List of Abbreviations vi

1 Introduction 1

1.1 Contributions 2

1.2 Thesis Structure 3

2 Related Work 5

2.1 Diachronic semantic change detection 5

2.2 Time period and sources 6

2.3 Text Representations 7

2.3.1 Dimensionality reduction techniques 8

2.3.2 Neural network approaches 8

2.4 Types of similarity 9

2.4.1 Topical similarity 10

2.4.2 Attributional similarity 10

2.5 Time representation 11

2.6 Measures of similarity 13

3 Materials and Methods 17

3.1 Data 18

3.2 Methods 19

3.2.1 Orthogonal Procrustes 19

3.2.2 Incremental Training 21

3.2.3 Temporal Word Embeddings with a Compass 21

3.2.4 Pretrained Embeddings 22

3.2.5 LDA2vec 23

3.3 Evaluation 23

4 Experimental Results 25

4.1 Alignment methods 30

4.2 Pre-trained Embeddings 32

4.3 Lda2vec and Wordvec 33

5 Conclusion and Future Work 37

List of Tables

3.1 Corpus pair analysis [42]. 18

3.2 Pretrained embeddings. 22

3.3 Evaluation data overview 23

4.1 Results of the utilized models across different languages. 28

4.2 Top 5 teams from SemEval 2020 Task-1. 30

List of Figures

2.1 Workflow of Diachronic Semantic Change Detection 6

2.2 PCA visualisation of the usage representations of the word "atom". [12] 15

2.3 Probability-based usage type distributions of word "atom" across different decades [12] 16

3.1 A and B sets of points in x-y axis, before the application of OP [28]. 20

3.2 Points of orthogonal matrix RA and B after the application of OP [28]. 20

3.3 Incremental training 21

3.4 TWEC training. 22

4.1 Pipeline of the experiments for each different language. 26

4.2 Box-plot of F1 scores per alignment method 31

4.3 Box-plot of F1 scores per type of embeddings' initialization 32

4.4 Box-plot of F1 scores per representation method 34

List of Abbreviations

BoW	Bag of Words
BERT	Bidirectional Encoder Representations from Transformers
CBoW	Continuous Bag of Words
DDSC	Detection of Diachronic Semantic Change
INCR	Incremental Training
JSD	Jensen-Shannon Divergence
LDA	Latent Dirichlet Allocation
NN	Neural Network
OP	Orthogonal Procrustes
PCA	Principal Component Analysis
PMI	Pointwise Mutual Information
RI	Random Indexing
SGNS	Skip-gram with Negative Sampling
SVD	Singular Value Decomposition
TF-IDF	Term Frequency-Inverse Document Frequency
TWEC	Temporal Word Embeddings with a Compass

Chapter 1

Introduction

The work herein constitutes a comparative study of text representation methods on the task of *diachronic semantic change detection* of words. Our objective is to explore the related bibliography and apply or combine existing methods in order to examine their respective performance. Moreover, through our implementation, we aim to provide an adaptable system which incorporates the basic components of diachronic semantic change detection and could be leveraged from the research community in future works.

The work in [11] defines natural language as a discrete symbolic representation of human knowledge. The combinations of these symbols create words, sequences of the words form sentences, and sentences can lead to written or phonetic forms of human knowledge. Language drifts refer to the changes in these components. The different types can be classified into linguistic drifts and cultural shifts[18]. The first involves changes in the true sense of words, and the latter considers emerging cultural associations. A prominent example of linguistic drift is the word "villain", with an initial meaning of a man who works on a farm and its contemporary sense of a mean person or a criminal. On the other hand, an example of a cultural shift is the word "apple". Although it is still used as a kind of fruit, in the last decades, it also refers to the products of an international technology company.

Linguistic drifts can be further divided to *lexical*, where new words appear in the lexicon, *phonological*, involving any sound change on words, *spelling*, that relates to

orthographic aspects, and *semantic change*. Focusing on semantic change, we have the following classification:

- pejoration: negative meanings over positive
- amelioration: positive meanings over negative
- broadening: additional potential meanings
- narrowing: restricted potential meanings
- shift: original meaning is not available anymore
- differentiation: two meanings arise from a single original one

Semantic change detection provides the necessary tools to researchers from social and humanities sciences in order to identify shifts in language, and hence the changes that occur in society. Furthermore, semantic similarity is utilized from computational sciences in several other tasks. These kind of tasks include word sense disambiguation[34], text classification[29], sentiment analysis[1, 22], machine translation[55], question answering[24] and information retrieval[53].

1.1 Contributions

In this work, we deal with the task of diachronic semantic change detection between two different periods. The task is addressed as a binary classification problem depending on the indication of meaning change (true/false) for specific words. We examine the literature on the diachronic semantic change detection task, and we highlight the key-point decisions that have to be considered to deal with it. After that, we provide the methodology of our empirical study, which focuses on three different layers. The first lay on the *text representations*, where we compare the methods of Word2vec[25, 26] and LDA2vec[27] and we examine whether the "injection" of LDA[6] topics into the Word2vec manages to capture the semantic similarity of the words in a better way. Next, we explore pre-trained embeddings[30, 52, 50] in combination with Word2vec. In this case, we examine whether the exploitation of pre-trained embeddings leads to an increase in the average performance of the system. The second layer is about the different approaches to the notion of *time representation* of the word embeddings. We concentrate on the alignment methods

of Orthogonal Procrustes[28], Incremental Training[17] and Temporal Embeddings with a Compass[3], and we experimentally assess whether the alignment approach of the word embeddings affects the performance of Word2vec. Finally, for the third layer of similarity measures, we apply the cosine distance and the local neighborhood measure[13] to all of the previously presented experiments. We investigate whether the local neighborhood measure affects the performance of LDA2vec and how it improves on the performance of the respective cosine distance measure.

1.2 Thesis Structure

The rest of the document is organized as follows. In Chapter 2 we present the work that is closely related to the task of unsupervised lexical semantic change detection. We present a workflow that is usually followed in this task and fit the related work within this workflow. Chapter 3 presents the methodology that we followed. We describe that data we used and the methods that we applied, leading to a series of experiments. Then, in Chapter 4 we present the different experimental results that emerged from the various settings of our system. We discuss how our results compare to the best results of the SemEval 2020 Task-1, and we further answer the research questions posed above. Finally, Chapter 5 concludes the thesis and presents steps and ideas for future work.

Chapter 2

Related Work

This chapter provides essential terms, background knowledge, and related work in diachronic semantic change detection. Furthermore, we present the critical decisions an individual has to take to handle the detection of diachronic semantic change (DDSC) on specific words. We break down and present the important aspects of the task, referring to the related literature respectively:

- Diachronic semantic change detection workflow [2.1]
- Time period and sources [2.2]
- Text representations [2.3]
- Similarity types [2.4]
- Time representation [2.5]
- Similarity measures [2.6]

2.1 Diachronic semantic change detection

The starting point of DDSC is to define the traits of the corpus/dataset under investigation. Two critical components are the time period that the data came from and the type of source that originated it. A corpus can be derived from literature, newspapers, online communities (chats, social media), reports, and more. Such types of sources have differences in the form of their language, the audience they refer to, and hence the degree of semantic changes in a specific time period[7]. Next, there is the choice of algorithm for the text representation. This decision lies on several

factors such as the required training time of our model, the type of similarity it captures[2.4], and the amount of the available data[10]. Given the diachronic nature of the task, a comparison of at least two different text representations, one for each of the earlier and later corpora, is needed. To obtain comparable word representations, we need them either to be trained in a common space from the start [17, 3] or to use an alignment method to map the independently trained representations in that common space[28]. Finally, we need to identify the suitable measures [2.4] to quantify the degree of change. The following sections is an elaboration on each of the workflow steps of the figure [2.1].

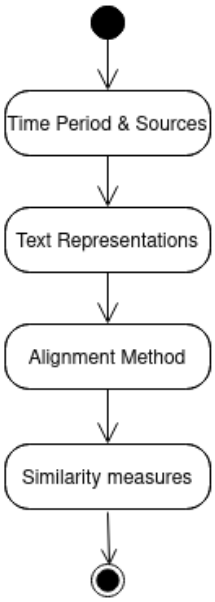

Figure 2.1: Workflow of Diachronic Semantic Change Detection

2.2 Time period and sources

The authors in [7, 23] examine semantic change in a short-term period, where text sources are discourses from online communities and news. Both types of sources are rapidly affected by cultural changes. Therefore linguistic changes are adopted and spread quicker. For example, words that refer to new emergent technologies or slang words from a viral song (cultural changes) are most likely to exist in a magazine or social media rather than in a book. On the other hand, in literature, there is a need for more persistent changes in terms of time dimension for a semantic change to be reflected in books. A substantial difficulty in short time periods is handling

referential cases. Referential cases emerge from words that, in certain periods, are closely related to specific facts, persons, etc. This might lead to high differences in the context of the word before and after that certain period. In turn, this leads to cases falsely recognized as semantic changes. It is, therefore, crucial to select an appropriate measure that handles such cases efficiently [7, 23].

2.3 Text Representations

In this work, we are interested in DDSC of specific words between two periods of time. Hence we create text representations (embeddings) at the word level. Embeddings can be derived from a document or a window of words; therefore, the word representation depends on the choice of context and vocabulary size.

Text representations can be separated into sparse and dense. The former is calculated from co-occurrence matrices and is based on raw count frequency and their relevant probabilities. Typical examples are the bag of words (BoW) model and the term frequency-inverse document frequency (tf-idf) weighting scheme. BoW results in a vector where rows stand for each different document and columns for the count of each unique term occurrence in a document. This approach results in high values for words with high frequency, such as conjunctions, even if they are not informative. Tf-idf solves this problem by considering the frequency of a term in all documents. The final vector contains low values for high-frequency words and high values for rare words.

Further calculations can be applied to the raw count frequency to avoid skewed distribution and retrieve the degree of information in each context-target word combination. This can be achieved with the usage of Pointwise Mutual Information (PMI) and its various forms (Positive PMI, Weighting PMI).

Dense or distributed representations aim to describe the meaning of words and sentences, solving the problem of the high number of features caused by sparse representations due to the vocabulary size. Dense representations are usually derived from either the application of dimensionality reduction techniques or a neural network (NN) architecture[30, 26].

2.3.1 Dimensionality reduction techniques

Among the most representative methods in dimensionality reduction are Principal Component Analysis (PCA), an application of Singular Value Decomposition (SVD) [47], and Random Indexing(RI) [40]. PCA is an unsupervised algorithm seeking a lower-dimensional space that maximizes the variance in a dataset. It first samples the data in a standard co-occurrence matrix and then transforms it into a much smaller and denser representation. RI is a random projection technique that lies on the Johnson-Lindenstrauss lemma [16]. The lemma states that high dimensional spaces can be approximately projected into lower dimensions preserving the distances between points. RI, compared to PCA, is computationally cheaper and more flexible in cases of newly appeared data [40].

2.3.2 Neural network approaches

NN approaches can be further divided into pre-trained contextual and non-contextual embeddings. Two prominent examples of the first case are the state-of-the-art ELMo[31] and the later method of Bidirectional Encoder Representations from Transformers (BERT) [8]. Both are pre-trained in massive volumes of textual data. BERT has two layers for the training phase. The first is where the model understands the language. The second is fine-tuning the model's last layers to manage a specific task, such as sentiment analysis, question answering, or machine translation. It learns the language model by simultaneously solving two unsupervised tasks, Masked Language Model (MLM) and Next Sentence Prediction (NSP). In MLM the input is a sentence with certain masked words, and the output is the prediction of that words. In NSP the input is two sentences A and B, and the model predicts whether sentence B is followed by sentence A or not. The input sentence in both cases is represented as word embeddings calculated from the addition of three different embeddings; the token pre-trained embeddings from WordPiece[49], the segment embeddings which encode the number of the sentence, and the positional embeddings that express the position of a token in a sentence. BERT learns the context of a word based on all of its surroundings and can represent different senses

of the same word, e.g., "get", which might indicate the verb of "procure", "become", or "understand".

The well-known word embeddings that do not capture contextual information are the predictive model of Word2vec[26, 25], the count-based model of Glove[30] and FastText[2]. Word2vec learns to predict the context words from a pivot word (Skip-gram) or a pivot word from a given context (Continuous Bag of Words) using local window-based methods. Word2vec is a one hidden layer neural network that utilizes the methods of backpropagation and stochastic gradient descent to optimize the relevant weights. The input is a one-hot encoding vector, and the output is the probabilities distribution of a softmax function. The output layer is useful only for the training process since the method's purpose is to create word embeddings. The final word embeddings are represented by the weights of the hidden layer. On the other hand, Glove is based on leveraging the global word-to-word co-occurrences and the application of matrix factorization to obtain a dense representation of the words. Finally, FastText only differs from the Skip-gram method in the elements it considers as the context. Skip-gram uses the word n-gram as the context, while FastText utilizes character n-grams.

Being able to choose between the different distributional representation of words, the next step for the DDSC is to identify the type of similarity we want to capture, if any specific, and then choose the corresponding algorithm for the text representation.

2.4 Types of similarity

Relevant studies in the literature are grouped into two main categories: those that focus on topical similarity [19, 39, 15, 6] and those on attributional [17, 13, 14, 37, 36, 12]. Moreover, there are also works that have managed to leverage both types of similarity [27, 29].

The topical similarity is based on the assumption that words co-existing in similar topics tend to be more similar. Topic modeling is the suite of algorithms that aim to deal with the extraction from documents by the mathematical inspection of relationships between words and documents containing them. It interprets the textual world on the assumption that a document can refer to several topics and

each of these topics as a distribution over some vocabulary words.
On the other hand, attributional similarity [46] is about the degree of the shared features between the properties of two words. Words with the same properties can be considered synonymous. A way to represent these properties from a given corpus is by leveraging contextual similarity and dense representations[2.3.2].

2.4.1 Topical similarity

Latent Semantic Analysis (LSA)[19] is a foundational method in the field of distributional semantics and probably constitutes the beginning of topic modeling. It can be used to capture the topical similarity of target words. It is designed to learn in an unsupervised way term-term, term-document, and document-document concepts by leveraging tf-idf weights in combination with SVD. The result of LSA[1] is the production of term-topic and document-topic representations.
The authors in [39] moved one step beyond, adding the concept of context vectors[44] and assuming that these representations can capture the topic of a target word. In this case, Infomap[45] is used to initialize the vocabulary and discover terms.
Other approaches are Probabilistic Latent Semantic Analysis (pLSA)[15], which improves on SVD by adopting a probabilistic approach. An additional improvement relaxing the assumption of whether a document refers to a single topic or multiple topics was introduced by the generative probabilistic model of Latent Dirichlet Allocation (LDA)[6]. This approach was the Bayesian version of the pLSA. The final output of LDA is a document weight vector that contains the probabilities for the mixture of topics in each document and the probabilities of words that characterize each topic.

2.4.2 Attributional similarity

In the context of attributional similarity, much work applies a Skip-Gram model with negative sampling (SGNS) of more distant words [17, 13, 14]. Skip-gram is an NN approach, so the word embeddings result from training, comparing with the

[1] A detailed explanation of LSA can be found at 'Handbook of Latent Semantic Analysis',Thomas K. Landauer, Danielle S. McNamara, Simon Dennis, Walter Kintsch, 2007

"truth" (labeled data), and updating the weights. The negative sampling of more distant words is a technique with which, instead of updating all of the weights of the NN, it updates only those that positively affect the prediction of our target word/sentence and just a small percentage of the negative words. This technique optimizes the number of weights calculation and hence the overall need for resources. The authors in [17] managed to build one of the first approaches to automatic detection of semantic shifts across different periods by utilizing embeddings from SGNS. Moreover, the work in [14] used SGNS as one of the baselines in a work that introduced the empirical statistical laws of conformity and innovation. Conformity states that frequently used words change at slower rates, and innovation means that polysemous words change faster.

The combination of the time continuity concept with SGNS is the result of [37], in which instead of creating term-context embeddings, the authors created differentiable functions that return term-context embeddings at time t. BERT has been considered a baseline model in NLP works [36], including semantic similarity detection. Interesting approaches to further leveraging its performance have been proposed. The work in [29] proposes tBERT, a combination of BERT with LDA and Gibbs Sampling Dirichlet Multinomial Mixture (GSDMM), for the task of sentence pair classification. Topics improved overall BERT performance, especially on datasets with domain-specific words. Furthermore, the proposed model performed better than the longer fine-tuned BERT.

Finally, the authors in [12] provide an insight into detecting different usage types of words. They applied clustering to different words' senses and usages using BERT representations. According to the empirical study, this model can efficiently capture diachronic (narrowing, broadening) and synchronic (metaphor, polysemy, syntactic functionalities) linguistic phenomena.

2.5 Time representation

The detection of diachronic semantic change inherently carries the concept of comparison, and to make any comparison, we need comparable objects. This section presents some of the most common approaches to creating comparable objects in

the text representation. These depend either on transformations or methodological core assumptions. Dense representations of a target word in different periods, specifically those derived from sophisticated approaches such as SGNS and SVD, are not in the same semantic spaces. This happens due to the stochastic nature of these approaches. In this context, the work in [14] handles the problem of comparable objects creation with the usage of the Orthogonal Procrustes (OP) technique[43] for the alignment of different word embeddings. Procrustes solves the problem of the mapping between two different sets of points, e.g., s_1 and s_2, by adjusting vectors to fit the same coordinate axes. After applying this technique, two representations of the same word can be subject to comparison. For example, W_t for word embedding in time period t and W_{t+1} for time period t_{+1}.

There are cases, however, where the alignment is not applicable. In simple text representations such as BoW or PMI, where the degree of interpretability is much higher, different representations of the same word are inherently aligned. In other works, the idea of a common shared ground (weights initialization) upon training effectively manages to produce vector embeddings that lie in a common semantic space. Such examples are [17] and [3].

In particular, the authors in [17] followed an approach of incremental training through different time periods to achieve comparable text representation objects. With this approach, word embeddings of each time step t initialize the weights for calculating the word embeddings in time step *t+1*.

In the second example, the work in [3] proposed the entity of a compass, which reflects the embeddings derived from the training procedure on the whole corpus. These word vectors are then used to initialize the weights of each time period's vectors before their corresponding training. The concept of a compass lies in the shared vector initialization step among vectors of different time periods.

Other cases with no need for semantic space alignment are embeddings derived from context words; neighbor words within a predefined window. In [21], the authors used CBOW to create the context embeddings of each word for time windows of 20 years duration. The assumption here is that a word is represented by its neighbors. Hence for the task of DDSC they do not contrast word embeddings in different time periods,

but they compare the overlap of the neighbor words across time.

Finally, the work in [23] used another technique in which, for each token, time-specific representations were created by averaging all contextual embeddings within specific periods using BERT. This simplified the vector creation process for each word since BERT captures a different vector representation for any occurrence of the same word if this appears in different contexts.

2.6 Measures of similarity

Measuring similarity in text, or in other words, calculating the difference between two or more representations of a word, can be performed with either intuitive or more complicated metrics, and it depends on the assumptions of each experiment. Cosine distance is by far the most widespread of those measures. It is used in cases of vectors' comparison, and it is a way to measure the degree of proximity in multidimensional spaces. The formula for the cosine similarity of two vectors is:

$$Cosine(x, y) = \frac{x \cdot y}{|x||y|} \tag{2.1}$$

In this formula, the dot product of x and y vectors is divided by the product of their relevant magnitudes. Conceptually, it is the cosine of the angle between two different vectors, and this angle indicates, in a quantifiable way, their degree of similarity. We can use the cosine similarity measure to compare sentences, words in different time periods, or pairs of words.

The Jaccard coefficient, another measure, counts the similarity between two different sets, U and V. The formula is:

$$Jaccard(U, V) = \frac{|U \cap V|}{|U| + |V| - |U \cap V|} \tag{2.2}$$

The Jaccard coefficient divides the number of shared members among two different sets with the number of the total distinct members of the sets. The authors in [21] used the Jaccard coefficient to detect diachronic semantic drift. They proposed that the consistency of words' meaning can be reflected in the changes of their top k neighbor words across different periods. Diachronic semantic drift detection is

achieved by comparison on top neighbor words in time t and t_{+1}.

An interesting finding from the work of [7] was that the cosine distance could not capture referential cases efficiently. Their proposed method to deal with referential cases is the measure of *contextual variability*. This measure is calculated by the average pairwise cosine distance between context vectors of targeted words. The context is a window of five words around the word of interest, and the relevant vectors were computed by averaging their respective embeddings. The model used in that work was from [17].

Moreover, the authors in [13] proposed the local neighborhood measure (LNM) for the detection of meaning change due to cultural factors and the indication of the relevant period of shift. The intuition of this measure is that meaning shift can be reflected in the changes of the most similar words. In order to calculate it, we need to obtain a set S_N with the N most similar words of a pivot word w_i in time periods t and t_{+1}. After that, we create two vectors V_t, V_{t+1} with the cosine similarity of w_i and the respective set words of S at each time period. Finally, we compute the cosine distance between the vectors of the different time periods. The local neighborhood measure can be expressed by the following formulas:

$$V(w_i)_t = cos - sim(w_i, w_j) \forall w_j \in S_N \tag{2.3}$$

$$LNM = cos - dist(V(w)_i, V(w)_{i+1}) \tag{2.4}$$

In a different direction, the work in [41] utilized word entropy as a metric of metaphoric change detection in different time periods. This measure was applied to the respective normalized co-occurrence matrices. The normalization step was necessary to obtain the conditional probability distributions of context words given the target word.

Finally, the authors in [12] leveraged Jensen-Shannon Divergence (JSD) as a tool to indicate a semantic change in sense narrowing and broadening. For each word of interest they created a usage matrix U_w= (w_1,...,w_N) which contained a

set of their corresponding N usage representations. Each w_i representation consisted of the context of the word, a 128-token window, and a label t_w, which indicated the time period of the representation. The standardized values of U_w constituted the input in a K-Means model, and the number of clusters was determined with the usage of silhouette score [38]. The number of clusters for the usage matrix contents was the number of different types of usages for a target word. Below there is a depiction of the clusters of the word "atom".

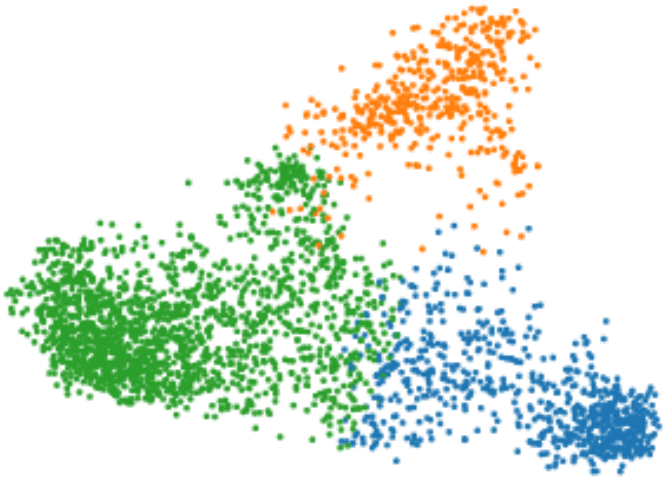

Figure 2.2: PCA visualisation of the usage representations of the word "atom". [12]

From the counts of cluster occurrences, they obtained frequency distributions for each time period that they transformed into probability distributions by normalization. At that point, JSD was an applicable measure since it is a way to compare different distributions. High JSD values implied a significant difference between distributions (increase-decrease in the number of clusters), while low values (same number of clusters) implied slight changes across time periods. The following figure depicts the probability distributions across different periods.

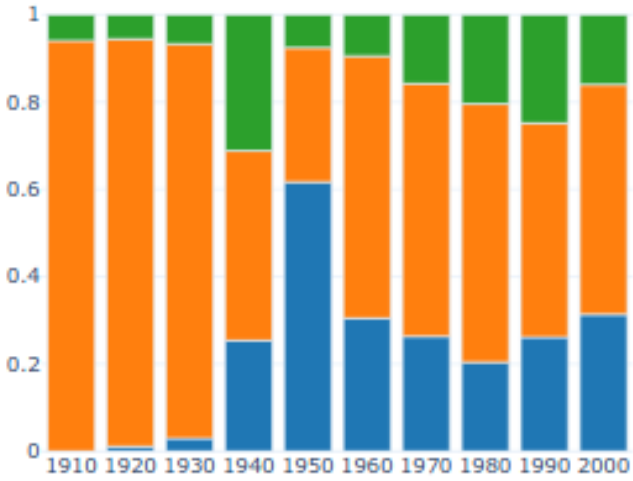

Figure 2.3: Probability-based usage type distributions of word "atom" across different decades [12]

Chapter 3

Materials and Methods

This chapter describes the core components of our methodology, the pipeline we follow, and the scientific questions we aim to answer through our experimental setting. We have experimented with various approaches with different parameters to examine their performance on the classification subtask of SemEval-2020 Task 1[42]. SemEval-2020 Task 1 concerns unsupervised lexical semantic change detection. Given subcorpora from two different time periods, participants are asked to submit their proposals on the subtasks of binary classification and ranking. In this work, we focus on binary classification and, in particular, whether a word's initial sense changed drastically or not between the two time periods.

In [2.5] we saw that the creation of comparable embeddings is a necessary step in DDSC. Hence, our first interest is to examine the different alignment methods and whether there is a method that improves the performance of our system. In our setup we examine the methods of Orthogonal Procrustes (**OP**) from [14], the incremental training (**INCR**) approach from [17], and the temporal word embeddings with a compass (**TWEC**) from [3].

The second question is related to pre-trained embeddings and whether the prior weights' initialization of a model manages to increase the overall performance. We used pre-trained embeddings which where calculated with the methods of Glove[30], Word2vec[26] and Wikipedia2Vec[50]. Finally, we investigated LDA2vec[27], which produces a different type of embeddings encapsulating topical and attributional similarity, to check if it overwhelms the widely adopted approach of Word2vec or

not.

Both the training and test datasets are provided from SemEval-2020 Task 1. The training data consist of four sets (earlier-later) of a corpus, one for each of the English, German, Swedish and Latin languages. The respective test data is a gold standard dataset for each language. The following sections provide the details of the core components of our methodology.

3.1 Data

As mentioned above, the training data are provided from SemEval-2020 Task 1. There are data for four different languages: English, German, Latin, and Swedish. For each language, the data are split into two distinct time periods (*C1* and *C2*). The provided datasets derived from an extraction on the original sources, for each of the different time periods. The range of the time periods was a choice based only on the data size and the availability of target words. All sentences with less than 10 tokens were removed, however, for the Latin the applied threshold was the number of 2 tokens. A downsampling technique was applied in corpora of C2 of German and English, in order to meet the data size of C1. For that purpose all sentences with target lemmas were kept and combined with a similar size of random sentences that did not contain any of them. The final form of the C1, C2 subcorpora was a result of lemmatization on the tokens, removal of the punctuation and at last random shuffling of the sentences. Table 3.1 provides an analysis of the corpora, which contains their sources, the period of their publication, the number of tokens, the number of distinct occurrences of the tokens (types), and finally, the type-token ratio (TTR) (number of types/number of tokens * 1000). The analysis table came from [42].

	C1					***C2***				
	corpus	**period**	**token**	**types**	**TTR**	**corpus**	**period**	**token**	**types**	**TTR**
English	CCOHA	1810–1860	6.5M	87k	13.38	CCOHA	1960–2010	6.7M	150k	22.38
German	DTA	1800–1899	70.2M	1.0M	14.25	BZ+ND	1946–1990	72.3M	2.3M	31.81
Latin	LatinISE	-200–0	1.7M	65k	38.24	LatinISE	0–2000	9.4M	253k	26.91
Swedish	Kubhist	1790–1830	71.0M	1.9M	47.88	Kubhist	1895–1903	110.0M	3.4M	17.27

Table 3.1: Corpus pair analysis [42].

3.2 Methods

The structure of our work consists of five different approaches. Four of them share common ground as they utilize CBoW and SGNS in the context of the Word2vec model. They either deal with the semantic space alignment task [3.2.1], or the embeddings initialization technique [3.2.2], [3.2.3] which in our cases lead to common semantic space representations. The last approach is a combinatory method which leverages the capabilities of Word2vec at capturing attributional similarity[2.4.2] and LDA at capturing topical similarity[2.4.1].

In the sections below, we discuss these aspects. We start with the alignment method of orthogonal procrustes for word embeddings [3.2.1]. Then, we continue with the alternative method of incremental training that does not require alignment in a common space [3.2.2]. In the same spirit, [3.2.3] discusses temporal embeddings that also do not require an additional alignment. [3.2.4] discusses the use of pre-trained embeddings, and finally section [3.2.5] provides information about LDA2Vec for jointly learning word embeddings and topic representations.

3.2.1 Orthogonal Procrustes

Orthogonal Procrustes (OP), which has already been presented in [2.5], is a mathematical approach that we use on different embeddings when they are trained independently in a stochastic way and therefore cannot be compared directly. Therefore, we need such a method to bring them into a common space and transform them into comparable objects. This transformation is obtained by either rotating or resizing one of the embeddings to match the points of the other. Figure [3.1] presents a plot of two different sets of points on x and y-axis. We can think of them as values of different embeddings. For demonstration purposes of the OP method, the points are not random, and it is obvious that each set forms the same shape in different coordinates. After applying OP on these two sets, the result is presented in Figure [3.2] where the set A is rotated and resized, producing an orthogonal matrix RA, in such a way to minimize the distance from the points of set B. At this point, we can apply algebraic operations to calculate the distances of target words across

embeddings.

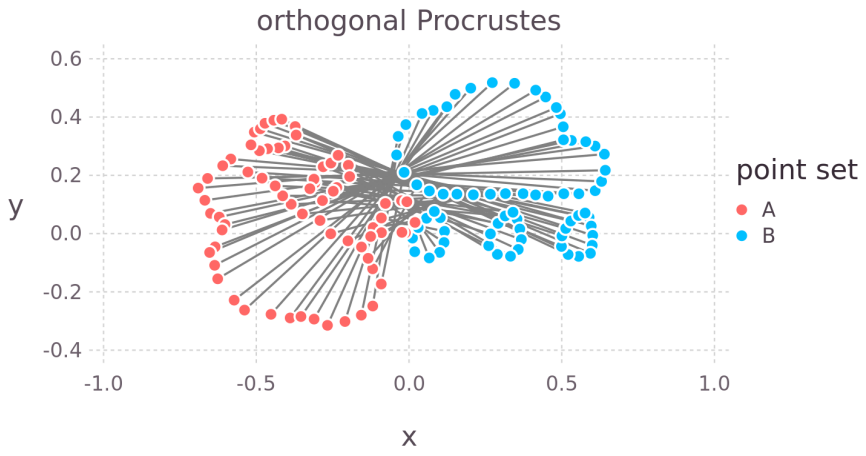

Figure 3.1: A and B sets of points in x-y axis, before the application of OP [28].

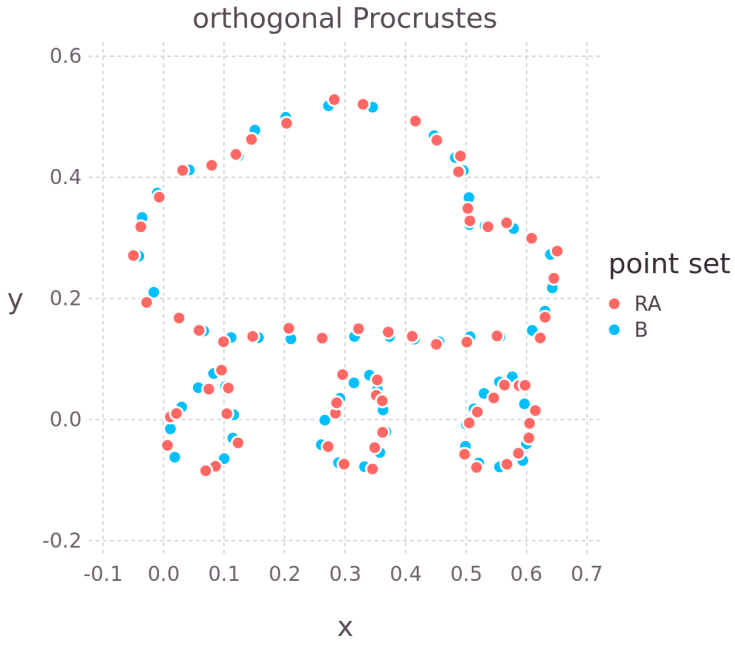

Figure 3.2: Points of orthogonal matrix RA and B after the application of OP [28].

3.2.2 Incremental Training

Incremental Training has been proposed in [17]. The goal is to create word embeddings in a common semantic space without needing an alignment tool, such as OP, presented in the previous section. To achieve this, we can initialize the word vectors in the second period with the weight learned during the training phase of the first period. The following Figure provides a visual representation of the approach and how we applied it in our case.

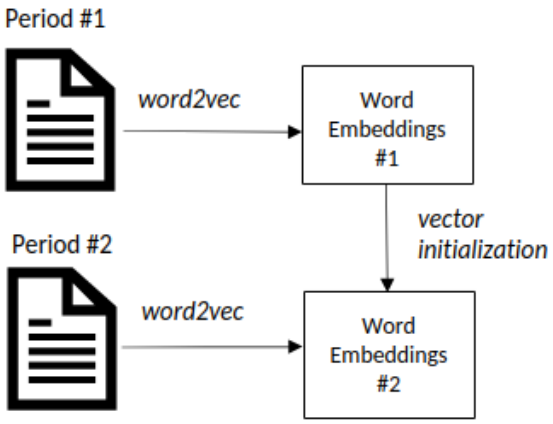

Figure 3.3: Incremental training

We train word embeddings on corpus $C1$ using the Word2vec method. The word embeddings of corpus $C2$ are initialized from the weights of $C1$. Then, the embeddings are updated during the training phase on $C2$. The incremental concept lies in the fact of the weights' update through the passage of time.

3.2.3 Temporal Word Embeddings with a Compass

Temporal Word Embeddings with a Compass (TWEC) is another technique, introduced in [3], for the creation of word embeddings in a common semantic space. It is very similar to the incremental training of [17] since they share the notion of creating word embeddings by prior weights initialization. The main difference is that in [17] the training procedure reflects the time passage from W_{t1} to W_{t2} and W_{ti} , where W_t is the word embeddings of each time period, while in [3] the word embeddings emerged from a single starting point, *compass*, which is the calculated embeddings from the corpora across different periods. In TWEC, each time period's

word vectors are the result of Word2vec training on the corresponding part of corpora (*corpus i*) and their vectors' initialization from *compass*. The following figure presents the high-level architecture of the TWEC training process.

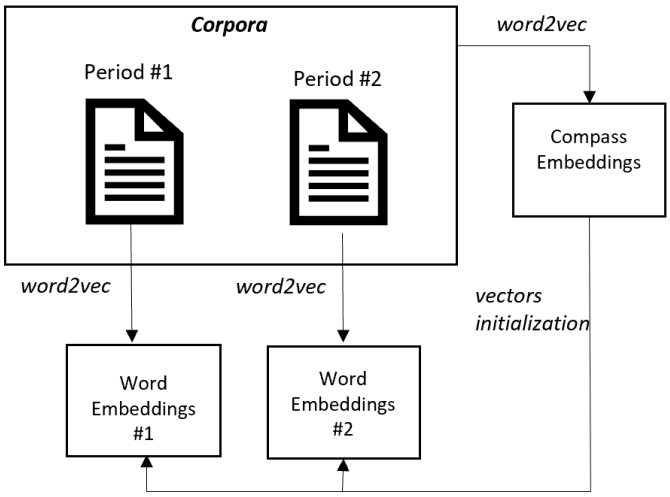

Figure 3.4: TWEC training.

3.2.4 Pretrained Embeddings

Pre-trained embeddings result from training on large-scale corpora. They offer a tool to individuals and researchers to perform algebraic operations on word vectors and finally deal with semantic similarity tasks. In our case, we used the following pre-trained vectors to examine how they affect the performance of the classification task. We used embeddings from three different sources, Glove [30] for English, Wikipedia2Vec [50] for German, CoNLL17 [52] for Latin and Swedish. The following table summarizes the information regarding the pre-trained vectors we used.

	name	**corpus**	**dim**	**window**	**vocab**	**method**
English	Glove	Wikipedia 2014, Gigaword 5	100	10	400K	Glove
German	Wikipedia2Vec	Wikipedia 2018	100	5	3.6B	Wikipedia2Vec
Latin	CoNLL17	CommonCrawl 2008-2017	100	10	555K	Word2Vec
Swedish	CoNLL17	CommonCrawl 2008-2017	100	10	3M	Word2Vec

Table 3.2: Pretrained embeddings.

3.2.5 LDA2vec

LDA2vec is a model proposed in [27] that leverages Word2vec ability to capture word associations based on a local window and LDA topics which capture global associations of words from a set of documents. The final word representations of the LDA2vec method are the result of the sum between the word embeddings, which are extracted with the SGNS method, and a document vector. The document vector, in accordance with LDA, consists of a document weight vector which contains the proportion of each topic in a document and a topic matrix. The topic matrix is a distributed representation of the topics that have been trained in a common space with the word vectors. LDA2vec produces jointly learned word embeddings, document representations, and topic representations. In our experiment, we utilized LDA2vec to create the embeddings of the two different periods.

3.3 Evaluation

Our evaluation setting for the models and the system we implemented is based on the gold standard dataset of SemEval-2020 Task 1. For each language, a different set of target words is provided with their relevant binary classification label. The label is a result of approximately 100k human judgments on whether a word meaning changed or not across the two different time periods C_1, C_2.
In the following table, we present an overview of the evaluation data, including the target words and their corresponding labels.

	C1 **token**	***C2*** **token**	**changed**	**stable**	**total**	**changed ratio**
English	6M	6M	16	21	37	0.432
German	70M	72M	17	31	48	0.354
Latin	1.7M	9.4M	26	14	40	0.65
Swedish	111M	71M	8	23	31	0.258

Table 3.3: Evaluation data overview

Binary Classification Rule

For the indication of the meaning change we have a applied a rule based on the notion of the standard error. In more detail, after obtaining our comparable text representations, we calculate cosine distances for all of the common vocabulary words in the different time periods. Hence, using that list of distances we compute the mean cosine distance and the respective standard error. Finally, we label a prediction as semantic change only if the cosine distance of a target word is greater than the sum of the mean cosine distance and the standard error. Given a common vocabulary V_{common} for two different time periods t and t_{+1}, we can define the threshold distance as

$$dist - list = cos - dist(w_t, w_{t+1}) \forall w \in V_{common} \tag{3.1}$$

$$dist - threshold = mean(dist - list) + stde(dist - list) \tag{3.2}$$

Chapter 4

Experimental Results

This chapter describes the experimental setup, the conducted experiments, and the results based on the methodology we followed. We have adopted a decoupled approach, separating the algorithmic implementations of the models, the evaluation measures, and the actions that can be chosen in our pipeline. This approach contributes to a quick and efficient reproduction of the experiments via bash scripts. It allows easy adaptations and extensions with new models for calculating word embeddings and new evaluation measures. In other words, we have created a pool of methods and steps that can be combined into a single pipeline of actions. These actions depend on whether we want to train embeddings, use pre-trained ones, use alignment methods, or use a different approach that learns representations in a common semantic space. Our system and implementations are publicly available[1], aiming to facilitate research in the field and provide a baseline implementation in an easy-to-use and understandable way to the research community. The following diagram describes the pipeline of our experiments.

[1] https://github.com/ichristod/language-drift

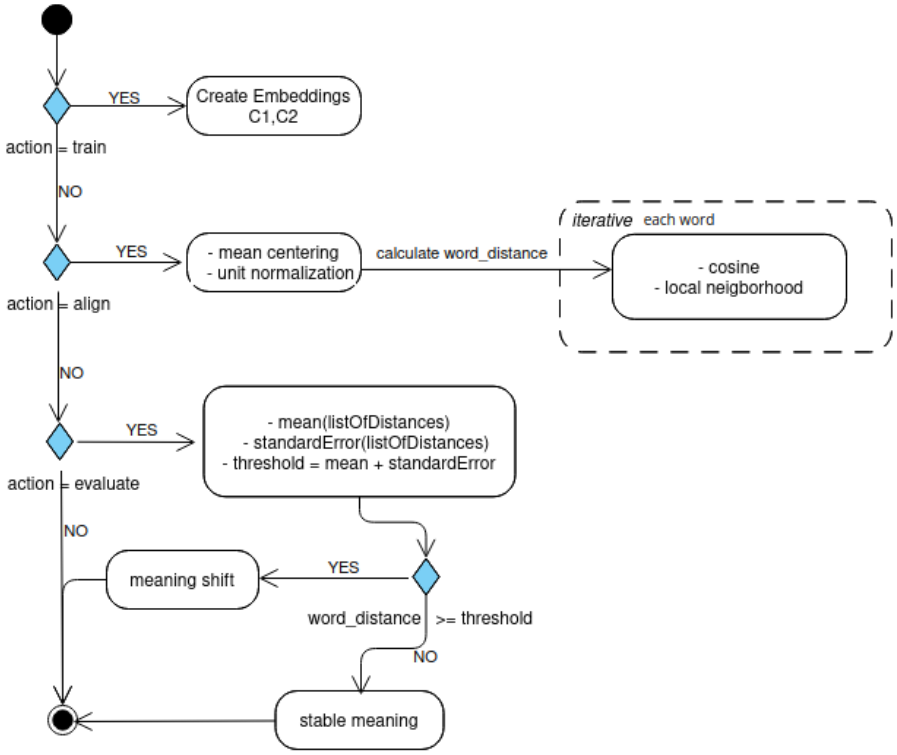

Figure 4.1: Pipeline of the experiments for each different language.

Common ground across executions

Our experiments comprise some common aspects. These are the datasets used, the settings of the Word2vec training process, the pre-processing steps, the alignment method, and the applied similarity measures. We have experimented in four languages: English, German, Latin, and Swedish. The creation of Word2vec representations is based on the wordvec model of the Gensim python library [35], and they share the same parameter settings. In more detail, for the Word2vec, we used:

- *training algorithms*: CBoW, Skip-gram
- *embeddings size*: 100
- *window size*: 10
- *minimum occurrence of words*: 3
- *negative sampling*: 3
- *sub-sampling threshold*: 1×10^{-3}
- *epochs*: 5

We have not performed an exhaustive search in the parameter space. However, the parameters above appeared good fits after grid searching. Moreover, before the

alignment, mean centering and normalization to unit length (L2 normalization) were applied to all trained embeddings according to [14, 20].

Conceptually, we are interested in the relations of words' embeddings, e.g., similarity, hence the direction of the vectors, and not the actual values[48]. Finally, we evaluate results against the gold standard dataset using two different similarity measures: cosine distance and local neighborhood measure.

Executions Results

The following table presents the results of all the experiments conducted, given all combinations of methods, embeddings, and measures. We have grouped the distinct executions based on the representation method, the alignment approach, the usage of pre-trained embeddings and the similarity measure. The applied performance measure is F1 score, which is expressed by the harmonic mean of precision and recall. In our case, precision is the number of the correctly predicted word with a change of meaning (true positives), divided by the total number of the words that were indicated as semantic change (true positives + false positives). On the other hand, recall, is defined by the fraction of the correctly predicted words that were labeled as a change (true positives), and, the number of the words that should be flagged as a meaning change (true positives + false negatives). Finally F1 score is calculated from the following formula:

$$F1score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{4.1}$$

Embeddings method	Alignment	Pre-trained	Similarity measure	LAT	EN	DE	SWE	Avg.
Skip-gram	Procrustes	true	Cosine distance	0.769	0.604	0.683	0.480	0.634
Cbow	Procrustes	true	Cosine distance	0.780	0.604	0.500	0.500	0.596
Skip-gram	Incremental	true	Cosine distance	0.788	0.604	0.548	0.410	0.588
Lda2vec	Procrustes	false	Local neighborhood	0.609	0.556	-	-	0.583
Cbow	Incremental	true	Cosine distance	0.800	0.604	0.523	0.353	0.570
Skip-gram	Procrustes	false	Local neighborhood	0.612	0.421	0.612	0.438	0.521
Skip-gram	Procrustes	true	Local neighborhood	0.680	0.444	0.578	0.370	0.518
Skip-gram	Incremental	true	Local neighborhood	0.667	0.526	0.542	0.333	0.517
Cbow	Twec	false	Cosine distance	0.679	0.500	0.389	0.485	0.513
Skip-gram	Twec	false	Cosine distance	0.625	0.615	0.468	0.312	0.505
Skip-gram	Incremental	false	Local neighborhood	0.667	0.595	0.444	0.296	0.501
Cbow	Procrustes	true	Local neighborhood	0.612	0.526	0.512	0.333	0.496
Cbow	Twec	false	Local neighborhood	0.681	0.438	0.522	0.333	0.494
Skip-gram	Procrustes	false	Cosine distance	0.667	0.457	0.449	0.370	0.486
Cbow	Incremental	false	Cosine distance	0.625	0.591	0.523	0.200	0.485
Skip-gram	Twec	false	Local neighborhood	0.571	0.500	0.565	0.296	0.483
Cbow	Procrustes	false	Cosine distance	0.604	0.410	0.524	0.387	0.481
Cbow	Procrustes	false	Local neighborhood	0.615	0.400	0.478	0.424	0.479
Cbow	Incremental	true	Local neighborhood	0.600	0.564	0.468	0.167	0.450
Cbow	Incremental	false	Local neighborhood	0.627	0.421	0.400	0.345	0.448
Skip-gram	Incremental	false	Cosine distance	0.356	0.512	0.523	0.387	0.445
Lda2vec	Procrustes	false	Cosine distance	0.200	0.222	-	-	0.211

Table 4.1: Results of the utilized models across different languages.

Initially, by examining the overall experimental results, we observe the following:

- On average, the use of skip-gram pre-trained embeddings that are further tuned separately on each time period and then aligned by the OP method seems to perform better than the rest of the methods when the cosine distance is used to assess the similarity between word vectors. This setup achieved an F1 score of 0.634.

- By looking separately at each dataset, we observe that on the Latin dataset, the use of CBOW pre-trained embeddings that are tuned in a common semantic space performs best, achieving an F1 score of 0.8. Regarding the English dataset, the method with the best results is TWEC, relying on Skip-gram embeddings that are trained from scratch directly on the used dataset. For

the German dataset, the best setting is that of using Skip-gram pre-trained embeddings that are further tuned and aligned with the OP method. This particular setting scores 0.683 in terms of F1. Finally, for the Swedish dataset, the best results are reached when using CBOW pre-trained embeddings that are further tuned and aligned with the use of the OP method.

Besides the aforementioned initial observations, we aimed at comparing the best systems submitted in SemEval 2020 to the ones proposed in this work. The table below presents the top five results achieved by the systems that were submitted in the SemEval 2020 Task-1. Comparing our results with those, our best model `(SGNS+PRE-TRAINED+OP+CD, F1=0.634)` is ranked at the second place. The best system from the submitted efforts was from [54], which achieved an F1-score of 0.646. Their approach was a combination of SGNS, Temporal Referencing[9] for the alignment of the embeddings, and finally the use of Gamma Quantile Threshold on the cosine distances as the classification rule.

Furthermore, comparing the results of SemEval 2020 Task-1 with our best system that did not utilize pretrained embeddings `(SGNS+OP+LNM, F1=0.521)`, we observe that it would be ranked in the fifth place.

An important observation is that two of our system variations, the one using LDA2vec representations and the one using TWEC as the alignment method, were not explored in the context of SemEval. The best settings of TWEC `(CBOW+TWEC+CD, F1=0.513)` and LDA2vec `(LDA2vec+OP+LNM, F1=0.583)`[2] approach would also be ranked in the fifth place.

Finally, what we overall observe, despite the absolute ranks in the context of SemEval, is that all results, including the ones produced in this work, are very close and comparable. In some cases, variations of our system achieved better scores in specific datasets. For instance, the best F1-score for the Latin dataset that our system reached was 0.8, while the best system in SemEval scored 0.769. Similarly, for English our best result was 0.615, while the best SemEval result was 0.58. The differences in the resuslts among datasets, both in our case and in SemEval

[2]For the verification of the results, LDA2vec needs also to be applied in German and Swedish languages. Preliminary experiments show that the results are close to the ones reported for other languages. However, due to hardware limitations, extensive experimentation is left for future work.

submissions, indicate that we probably need different methods for each language and suggest further experimentation.

Team	LAT	EN	DE	SWE	Avg.
Jiaxin-Jinan[54]	0.769	0.58	0.683	0.552	0.646
Skurt[5]	0.735	0.556	0.588	0.64	0.63
UWB[33]	0.727	0.588	0.7	0.5	0.629
UG Student Intern[32]	0.608	0.529	0.683	0.609	0.607
IMS[51]	0.608	0.514	0.634	0.710	0.5

Table 4.2: Top 5 teams from SemEval 2020 Task-1.

Besides the above initial observations, we further examine whether a particular alignment method performs significantly better, whether pre-trained embeddings improve the performance, and whether LDA2vec and Word2vec word representations perform the same. We aim to answer these questions in the following sections.

4.1 Alignment methods

Which vectors' alignment method performs better?

The first question we aim to answer is whether there is an alignment method out of those considered in our arsenal that performs better. Thus, our first experiment compared different alignment methods according to their respective model's performance. For the comparison, we used the three methods of **OP**, **INCR** and **TWEC**. The embeddings in each case were calculated after training with the Word2vec method. The null hypothesis was that all under investigation methods perform equally on the same datasets and parameter settings. To examine that, we ran 12 different executions for each alignment method and applied the two measures of similarity to them. Given the four different datasets, this approach resulted in 48 samples of F1 scores.

$$\overbrace{\underbrace{4}_{\text{languages}} \times \underbrace{2}_{\text{training algorithms}} \times \underbrace{3}_{\text{alignment methods}} \times \underbrace{2}_{\text{measures}}}^{\text{Number of samples}} = 48$$

The figure below presents the evaluation results in terms of the F1 scores of the three alignment methods using box plots, showing average, minimum and maximum values achieved, as well as outliers.

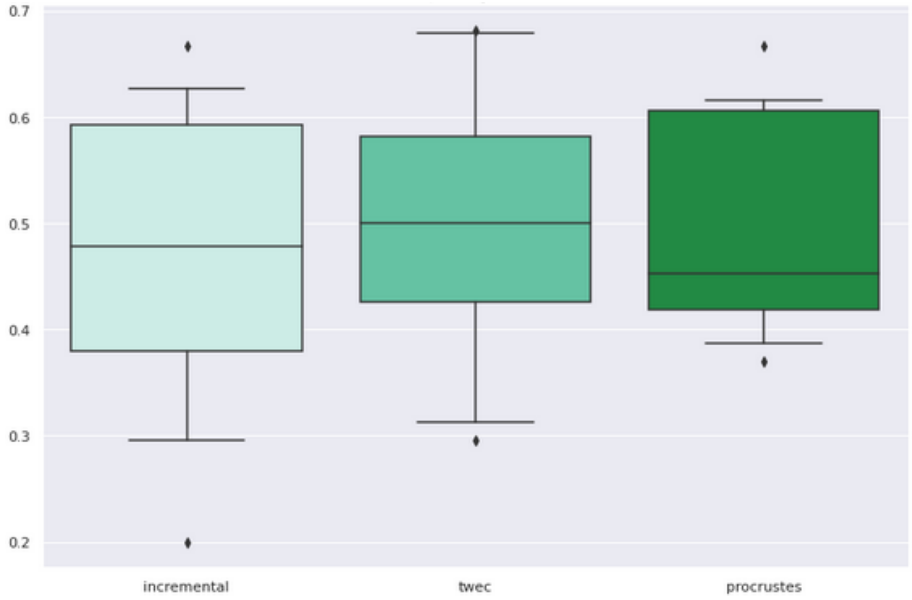

Figure 4.2: Box-plot of F1 scores per alignment method

To statistically test the differences, we applied the normality test of Shapiro-Wilk. Since OP didn't meet the criteria of normality, we continued with the non-parametric test of Kruskal Wallis. We failed to reject the null hypothesis, with a p-value of 0.81, indicating that the sets of samples had the same distribution. Next, we conducted the Wilcoxon Signed-Rank test for each pair (OP-INCR, INCR-TWEC, etc.). The latter is a non-parametric test for paired samples of two populations. We consider our samples as paired since the only variable that alternates is the one under investigation, i.e., the alignment method. Again, the null hypothesis was not rejected for all of the combinations. We, therefore, conclude that **there are no significant differences** between the alignment methods we explored on the parameter setting and the datasets we used, although it seems that *TWEC* performs better on average.

4.2 Pre-trained Embeddings

Do pre-trained embeddings improve performance?

The second research question we investigate is whether using pre-trained embeddings on massive corpora improves the performance of the system. Thus, in this experiment, we test the performance of the models initialized with pre-trained embeddings compared to those whose embeddings' weights were initialized in a stochastic way. We chose the pre-trained embeddings independently from their training method, even though we have used three different methods, Glove, Word2vec, and Wikipedia2Vec. For the alignment of the embeddings, we used the methods of **OP** and **INCR**. These combinations resulted in a total number of 64 experiments.

$$\overbrace{\underbrace{4}_{\text{languages}} \times \underbrace{2}_{\text{usage of pretrained}} \times \underbrace{2}_{\text{training algorithms}} \times \underbrace{2}_{\text{alignment methods}} \times \underbrace{2}_{\text{measures}}}^{\text{Number of samples}} = 64$$

The box plots below present the F1 scores of the experiments above, comparing the models with pre-trained embeddings to those without. Evidently, when using pre-trained embeddings, the system performs better on average. Next, we investigate whether this performance is significantly better.

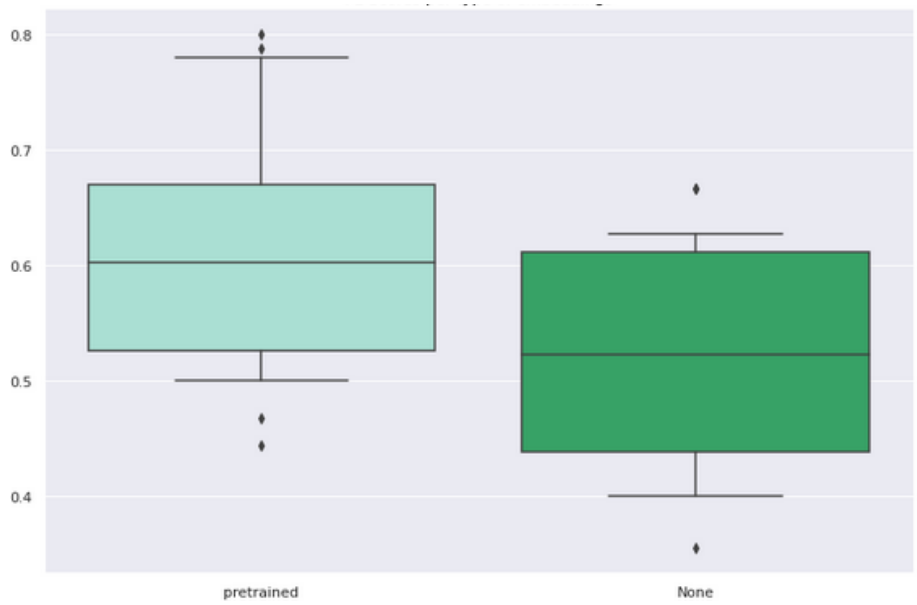

Figure 4.3: Box-plot of F1 scores per type of embeddings' initialization

In this case, the Shapiro-Wilk test indicated that both samples of the pre-trained

and non-pre-trained meet the criteria of normality. The paired t-test was our choice to check if there were differences between the distributions. One of the assumptions of this test is the absence of outliers. For the outlier detection, we applied the rule of the interquartile range (IQR). Therefore, we excluded samples located outside the range of 1.5xIQR. We identified two outliers on the Swedish dataset with this approach. The first was an execution of pre-trained embeddings, and the other was an execution without pre-trained. After removing outliers, we ended up with 62 samples in total. The null hypothesis of the paired t-test was rejected with a p-value of 0.02. Hence, we conclude that **the populations had statistically significant differences**. Comparing the means from pre-trained and non-pre-trained executions indicated an increase of 16% at the F1 scores when models are prior initialized with pre-trained embeddings.

4.3 Lda2vec and Wordvec

Do the representations of LDA2vec and Word2vec perform the same? The last research question concerns the application of different methods for creating word embeddings. In this last experiment, we explore the different types of word embeddings produced by LDA2vec to determine whether incorporating topical similarity in word embeddings could outperform our initial approach of Word2vec. For this experiment, we use executions that do not exploit pre-trained embeddings since LDA2vec is not applied in combination with pre-trained embeddings. Furthermore, the alignment method we used was the OP. The settings of the LDA2vec model are:

- *embeddings size*: 100
- *window size*: 10
- *minimum occurrence of words*: 3
- *negative sampling*: 3
- *topics*: 20
- *sub-sampling threshold*: 1×10^{-3}
- *epochs*: 5

The total number of the different LDA2vec samples was 4, since we tested one alignment method over the two different languages (English, Latin) and we used the two

measures of cosine distance and local neighborhood measure. We also managed to train two more models for German and Swedish, however, the respective aligned representations of the embeddings were not obtainable due to computational resources limitation[3].

$$\overbrace{\underbrace{2}_{\text{languages}} \times \underbrace{3}_{\text{training algorithm}} \times \underbrace{1}_{\text{alignment method}} \times \underbrace{2}_{\text{measures}}}^{\text{Number of samples}} = 12$$

Again, the box-plots of the F1 scores or the grouped executions of each representation method are presented below.

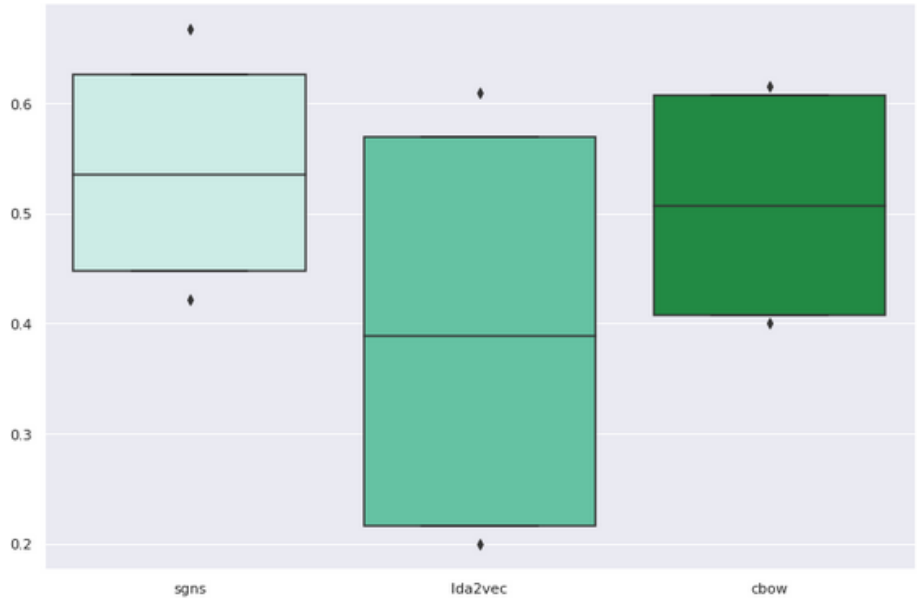

Figure 4.4: Box-plot of F1 scores per representation method

For the statistical tests, we followed the same steps as the previous experiments. We found that our distributions meet the criteria of normality through the Shapiro-Wilk test. Consequently, we moved on with the parametric test of paired t-test. The final results did not indicate differences between the LDA2vec and the pairs of SGNS and CBoW. However, through this experiment, we were able to identify that the local neighborhood measure seems more suitable for the LDA2vec since it resulted

[3]Memory requirements exceeded the capacity of our servers. However, preliminary results does not show any particular deviation from the values that are observed in the English and Latin datasets. Further experimentation to the much larger German and Swedish datasets is left for future work.

in an average F1 score of 0.583. In contrast, for the same setup, the cosine distance measure achieved an average F1 score of 0.211.

In summary, we investigated LDA2vec representation in comparison to the different methods of Word2vec and via our empirical evaluation we concluded that :

1. There are not significant differences between the performance of LDA2vec and Word2vec methods
2. Local neighborhood measure[13] seems more effective than cosine distance measure at capturing semantic similarity in LDA2vec representations.
3. LDA2vec in combination with local neighborhood measure, attained the highest average score between the models that utilized orthogonal procrustes without the usage of pre-trained embeddings.

Chapter 5

Conclusion and Future Work

In this thesis, we empirically studied the essential components of the workflow of diachronic semantic change detection. We presented a pipeline of steps and placed the related literature in this perspective, allowing a comprehensive understanding of the methods and techniques that are usually applied in the field under an unsupervised classification setting.

Moreover, via our comparative experimental setting, we explored different alignment methods and whether there is one that systematically achieves higher performance than the others. We concluded that there are no significant differences between the alignment methods of Orthogonal Procrustes, incremental training, and temporal word embeddings with a compass.

Aiming to explore the impact of different word representations, we also experimented with pre-trained embeddings versus embeddings trained directly in our datasets. We saw that the models with prior initialization of their embeddings perform better than those trained in our corpora from scratch. The increase in the F1 scores of the system when using pre-trained embeddings was at a percentage of 16%.

Finally, we also compared two different methods of text representation, Word2vec, and LDA2vec. LDA2vec combines both attributional and topical similarities. Our purpose here was to investigate whether a text representation, which incorporates LDA topics, was able to dominate the performance of the Word2vec. The results indicated that although Word2vec seems to perform better on average over all our datasets, there are no significant differences compared to LDA2vec. However, an

interesting finding is that in LDA2vec representations, the local neighborhood measure captures the semantic change in a better way than cosine distance manages to. This observation indicates that a model that considers topical similarities and neighboring words to create embeddings should be combined with a corresponding similarity measure that captures local similarities.

Future steps should focus on the application of our system in more datasets to ensure the robustness of the final results. Additional directions include approaching the task in a supervised learning way by training a system to classify word pairs with either similar meanings or not. Towards this direction, we aim to explore the application of supervised methods, such as K-nearest neighbors, as well as more sophisticated techniques, such as Siamese neural networks[4], that have the ability to learn from little data, and inherently, their goal is to learn a similarity function. Thus, they can be trained to see if two pairs of words are the same or not and therefore constitute a good fit for this task.

Bibliography

[1] Oscar Araque, Ganggao Zhu, and Carlos Iglesias. "A semantic similarity-based perspective of affect lexicons for sentiment analysis". In: *Knowledge-Based Systems* 165 (Dec. 2018). DOI: 10.1016/j.knosys.2018.12.005.

[2] Piotr Bojanowski et al. *Enriching Word Vectors with Subword Information*. 2017. arXiv: 1607.04606 [cs.CL].

[3] Valerio Di Carlo, Federico Bianchi, and Matteo Palmonari. *Training Temporal Word Embeddings with a Compass*. 2019. arXiv: 1906.02376 [cs.CL].

[4] Davide Chicco. "Siamese Neural Networks: An Overview". In: vol. 2190. Aug. 2020, pp. 73–94. ISBN: 978-1-0716-0825-8. DOI: 10.1007/978-1-0716-0826-5_3.

[5] Amaru Cuba Gyllensten et al. "SenseCluster at SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 112–118. DOI: 10.18653/v1/2020.semeval-1.12. URL: https://aclanthology.org/2020.semeval-1.12.

[6] A. Ng D. Blei and M. Jordan. "Latent Dirichlet allocation". In: *Journal of Machine Learning Research* (2003), 3:993–1022.

[7] Marco Del Tredici, Raquel Fernández, and Gemma Boleda. "Short-Term Meaning Shift: A Distributional Exploration". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics,

June 2019, pp. 2069–2075. DOI: 10.18653/v1/N19-1210. URL: https://aclanthology.org/N19-1210.

[8] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[9] Haim Dubossarsky et al. "Time-Out: Temporal Referencing for Robust Modeling of Lexical Semantic Change". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019. DOI: 10.18653/v1/p19-1044. URL: https://doi.org/10.18653%2Fv1%2Fp19-1044.

[10] Aysu Ezen-Can. *A Comparison of LSTM and BERT for Small Corpus*. 2020. DOI: 10.48550/ARXIV.2009.05451. URL: https://arxiv.org/abs/2009.05451.

[11] Lorenzo Ferrone and Fabio Massimo Zanzotto. "Symbolic, Distributed, and Distributional Representations for Natural Language Processing in the Era of Deep Learning: A Survey". In: *Frontiers in Robotics and AI* 6 (Jan. 2020). ISSN: 2296-9144. DOI: 10.3389/frobt.2019.00153. URL: http://dx.doi.org/10.3389/frobt.2019.00153.

[12] Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. "Analysing Lexical Semantic Change with Contextualised Word Representations". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 3960–3973. DOI: 10.18653/v1/2020.acl-main.365. URL: https://aclanthology.org/2020.acl-main.365.

[13] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. "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*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2116–2121. DOI: 10.18653/v1/D16-1229. URL: https://aclanthology.org/D16-1229.

[14] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. *Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change*. 2018. arXiv: 1605.09096 [cs.CL].

[15] Thomas Hofmann. "Probabilistic Latent Semantic Indexing". In: *the 22nd International Conference on Research and Development in Information Retrieval (SIGIR'99):1999* (Apr. 2004).

[16] William Johnson and Joram Lindenstrauss. "Extensions of Lipschitz maps into a Hilbert space". In: *Contemporary Mathematics* 26 (Jan. 1984), pp. 189–206. DOI: 10.1090/conm/026/737400.

[17] Yoon Kim et al. "Temporal Analysis of Language through Neural Language Models". In: *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*. Baltimore, MD, USA: Association for Computational Linguistics, June 2014, pp. 61–65. DOI: 10.3115/v1/W14-2517. URL: https://aclanthology.org/W14-2517.

[18] Andrey Kutuzov et al. *Diachronic word embeddings and semantic shifts: a survey*. 2018. DOI: 10.48550/ARXIV.1806.03537. URL: https://arxiv.org/abs/1806.03537.

[19] Thomas K. Landauer and Susan T. Dumais. "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge". In: *Psychological Review* 104.2 (1997), pp. 211–240. DOI: 10.1037/0033-295X.104.2.211.

[20] Omer Levy, Yoav Goldberg, and Ido Dagan. "Improving Distributional Similarity with Lessons Learned from Word Embeddings". In: *Transactions of the Association for Computational Linguistics* 3 (2015), pp. 211–225. DOI: 10.1162/tacl_a_00134. URL: https://aclanthology.org/Q15-1016.

[21] Ruiyuan Li, Pin Tian, and Shenghui Wang. "Study concept drift in 150-year english literature". English. In: *CEUR workshop proceedings* 2871 (2021). Publisher Copyright: © 2021 CEUR-WS. All rights reserved.; 1st Workshop on AI + Informetrics, AII 2021 ; Conference date: 17-03-2021, pp. 153–163. ISSN: 1613-0073.

[22] Youness Madani, Mohammed Erritali, and Bengourram Jamaa. "Sentiment analysis using semantic similarity and Hadoop MapReduce". In: *Knowledge and Information Systems* 59 (May 2019). DOI: 10.1007/s10115-018-1212-z.

[23] Matej Martinc, Petra Kralj Novak, and Senja Pollak. *Leveraging Contextual Embeddings for Detecting Diachronic Semantic Shift*. 2020. arXiv: 1912.01072 [cs.CL].

[24] Todor Mihaylov and Preslav Nakov. "SemanticZ at SemEval-2016 Task 3: Ranking Relevant Answers in Community Question Answering Using Semantic Similarity Based on Fine-tuned Word Embeddings". In: (2019). DOI: 10.48550/ARXIV.1911.08743. URL: https://arxiv.org/abs/1911.08743.

[25] Tomas Mikolov et al. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. arXiv: 1310.4546 [cs.CL].

[26] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL].

[27] Christopher E. Moody. *Mixing Dirichlet topic models and word embeddings to make LDA2vec*. 2016. arXiv: 1605.02019 [cs.CL].

[28] *orthogonal-procrustes*. URL: https://simonensemble.github.io/2018-10/orthogonal-procrustes.

[29] Nicole Peinelt, Dong Nguyen, and Maria Liakata. "tBERT: Topic Models and BERT Joining Forces for Semantic Similarity Detection". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 7047–7055. DOI: 10.18653/v1/2020.acl-main.630. URL: https://aclanthology.org/2020.acl-main.630.

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

[31] Matthew E. Peters et al. *Deep contextualized word representations*. 2018. arXiv: 1802.05365 [cs.CL].

[32] Martin Pömsl and Roman Lyapin. "CIRCE at SemEval-2020 Task 1: Ensembling Context-Free and Context-Dependent Word Representations". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 180–186. DOI: 10.18653/v1/2020.semeval-1.21. URL: https://aclanthology.org/2020.semeval-1.21.

[33] Ondřej Pražák et al. *UWB at SemEval-2020 Task 1: Lexical Semantic Change Detection*. 2020. DOI: 10.48550/ARXIV.2012.00004. URL: https://arxiv.org/abs/2012.00004.

[34] Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. "Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 99–110. URL: https://aclanthology.org/E17-1010.

[35] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[36] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. *A Primer in BERTology: What we know about how BERT works*. 2020. arXiv: 2002.12327 [cs.CL].

[37] Alex Rosenfeld and Katrin Erk. "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)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 474–484. DOI: 10.18653/v1/N18-1044. URL: https://aclanthology.org/N18-1044.

[38] Peter Rousseeuw. "Rousseeuw, P.J.: Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. Comput. Appl. Math. 20, 53-65". In: *Journal of Computational and Applied Mathematics* 20 (Nov. 1987), pp. 53–65. DOI: 10.1016/0377-0427(87)90125-7.

[39] Eyal Sagi, Stefan Kaufmann, and Brady Clark. "Tracing semantic change with Latent Semantic Analysis". In: *Current Methods in Historical Semantics*. Ed. by Kathryn Allan and Justyna A. Robinson. De Gruyter Mouton, 2011, pp. 161–183. DOI: doi:10.1515/9783110252903.161. URL: https://doi.org/10.1515/9783110252903.161.

[40] Magnus Sahlgren. "An Introduction to Random Indexing". In: *Language* (Jan. 2004), pp. 1–9.

[41] Dominik Schlechtweg et al. "German in Flux: Detecting Metaphoric Change via Word Entropy". In: *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 354–367. DOI: 10.18653/v1/K17-1036. URL: https://aclanthology.org/K17-1036.

[42] Dominik Schlechtweg et al. "SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 1–23. DOI: 10.18653/v1/2020.semeval-1.1. URL: https://aclanthology.org/2020.semeval-1.1.

[43] Peter Schönemann. "A generalized solution of the orthogonal procrustes problem". In: *Psychometrika* 31.1 (1966).

[44] Hinrich Schütze. "Automatic Word Sense Discrimination". In: *Computational Linguistics* 24.1 (1998), pp. 97–123. URL: https://aclanthology.org/J98-1004.

[45] Computational Semantics Lab at Stanford University's Center for the Study of Language and Information, eds. *Infomap NLP Software*. URL: http://infomap-nlp.sourceforge.net/.

[46] Peter D. Turney. "Similarity of Semantic Relations". In: *Computational Linguistics* 32.3 (Sept. 2006), pp. 379–416. ISSN: 1530-9312. DOI: 10.1162/coli.2006.32.3.379. URL: http://dx.doi.org/10.1162/coli.2006.32.3.379.

[47] Michael Wall, Andreas Rechtsteiner, and Luis Rocha. "Singular Value Decomposition and Principal Component Analysis". In: *In A Practical Approach to Microarray Data Analysis* 5 (Sept. 2002). DOI: 10.1007/0-306-47815-3_5.

[48] Benjamin J. Wilson and Adriaan M. J. Schakel. *Controlled Experiments for Word Embeddings*. 2015. DOI: 10.48550/ARXIV.1510.02675. URL: https://arxiv.org/abs/1510.02675.

[49] Yonghui Wu et al. *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016. DOI: 10.48550/ARXIV.1609.08144. URL: https://arxiv.org/abs/1609.08144.

[50] Ikuya Yamada et al. "Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, 2020, pp. 23–30.

[51] Frank D. Zamora-Reina and Felipe Bravo-Marquez. "DCC-Uchile at SemEval-2020 Task 1: Temporal Referencing Word Embeddings". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 194–200. DOI: 10.18653/v1/2020.semeval-1.23. URL: https://aclanthology.org/2020.semeval-1.23.

[52] Daniel Zeman et al. "CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies". In: *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 1–19. DOI: 10.18653/v1/K17-3001. URL: https://aclanthology.org/K17-3001.

[53] Shuo Zhang and Krisztian Balog. "Ad Hoc Table Retrieval using Semantic Similarity". In: *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*. ACM Press, 2018. DOI: 10.1145/3178876.3186067. URL: https://doi.org/10.1145%2F3178876.3186067.

[54] Jinan Zhou and Jiaxin Li. "TemporalTeller at SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection with Temporal Referencing". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, Dec. 2020, pp. 222–231. DOI: 10.18653/v1/2020.semeval-1.27. URL: https://aclanthology.org/2020.semeval-1.27.

[55] Will Y. Zou et al. "Bilingual Word Embeddings for Phrase-Based Machine Translation". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1393–1398. URL: https://aclanthology.org/D13-1141.