

REPUBLIC OF TÜRKİYE
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

**CROSS MODEL ALIGNMENT IN NATURAL LANGUAGE
PROCESSING**

Kadir GÜNEL

DOCTOR OF PHILOSOPHY THESIS
Department of Computer Engineering
Program of Computer Engineering

Supervisor
Prof. Dr. Mehmet Fatih AMASYALI

January, 2024

REPUBLIC OF TÜRKİYE
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

**CROSS MODEL ALIGNMENT IN NATURAL LANGUAGE
PROCESSING**

A thesis submitted by Kadir GÜNEL in partial fulfillment of the requirements for the degree of **DOCTOR OF PHILOSOPHY** is approved by the committee on 18.01.2024 in Department of Computer Engineering, Program of Computer Engineering.

Prof. Dr. Mehmet Fatih AMASYALI
Yildiz Technical University
Supervisor

Approved By the Examining Committee

Prof. Dr. Mehmet Fatih AMASYALI, Supervisor
Yildiz Technical University

Prof. Dr. Songül VARLI, Member
Yildiz Technical University

Prof. Dr. Tunga GÜNGÖR, Member
Bogazici University

Assoc. Prof. Dr. Ali Can KARACA, Member
Yildiz Technical University

Assoc. Prof. Dr. Murat Can GANİZ, Member
Marmara University

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Cross Model Alignment in Natural Language Processing supervised by my supervisor, Prof. Dr. Mehmet Fatih AMASYALI. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Kadir GÜNEL

Signature

Dedicated to my dear family



ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Prof. Dr. Mehmet Fatih Amasyali, my thesis advisor, for their invaluable guidance, patience, and immense knowledge. Their insightful feedback pushed me to sharpen my thinking and brought my work to a higher level.

I would also like to thank , Prof. Dr. Songül Varlı and Prof. Dr. Tunga Güngör for their expert advice and valuable suggestions during the course of this research.

I cannot forget to acknowledge the unwavering support and patience of my family who have been my constant source of encouragement and motivation. Their love and belief in me have been the bedrock of my perseverance.

Lastly, I would like to express my gratitude to all those who were involved in any capacity in this research or the preparation of this manuscript, and to those who provided emotional and practical support throughout my studies.

Kadir GÜNEL

TABLE OF CONTENTS

LIST OF ABBREVIATIONS	viii
LIST OF FIGURES	ix
LIST OF TABLES	xi
ABSTRACT	xii
ÖZET	xiv
1 INTRODUCTION	1
1.1 Structure of This Thesis	2
1.2 Impact of This Research	2
2 LITERATURE REVIEW	4
2.1 Evolution of Word Embedding Alignment	4
2.1.1 Bilingual Word Embeddings	5
2.1.2 Multilingual Word Embeddings	8
2.2 Sentence Embedding Techniques	8
3 WORD EMBEDDING ALIGNMENT	11
3.1 Point Cloud Registration	12
3.1.1 Iterative Closest Point	13
3.1.2 Least Squares & Orthogonal Procrustes Problem	14
3.2 ICP for Unsupervised Word Embedding Alignment: <i>vecmap</i>	15
3.2.1 Initialization	16
3.2.2 Training	16
3.3 Other Metods for Alignment	17
3.3.1 Optimal Transport	18
3.3.2 Generative Adversarial Networks	19
3.3.3 Gaussian Mixture Model	19
3.4 Similarity Searching: <i>KNN</i> vs. <i>CSLS</i>	20
3.5 Topological View of Alignment Models	21
3.5.1 Mesh Topology	22

3.5.2	Ring Topology	22
3.5.3	Star Topology	23
4	CROSS MODEL WORD EMBEDDING ALIGNMENT	24
4.1	Overview of Problem	24
4.2	Domain Adaptation Techniques for Cross Model Alignment	25
4.2.1	Correlation Alignment (CORAL)	26
4.2.2	Maximum Mean Discrepancy (MMD)	27
4.3	Using Pivot Language for Cross Model Alignment	28
4.3.1	CORAL	30
4.3.2	MMD	31
5	CROSS MODEL KNOWLEDGE DISTILLATION IN SENTENCE EMBEDDING MODELS	32
5.1	Cross Model Alignment with Knowledge Distillation	32
5.2	Methodologies	33
5.2.1	Distance Minimization	34
5.2.2	Distance & Perplexity Minimization	35
6	EXPERIMENTS	37
6.1	Experiments on Cross Model Alignment in Word Embedding Models	37
6.1.1	Data	37
6.1.2	Experiment Procedure	39
6.2	Experiments on Cross Model Knowledge Distillation in Sentence Embedding Models	41
6.2.1	Data	41
6.2.2	Experiment Procedure	44
7	RESULTS	46
7.1	Results for Cross Model Alignment in Word Embedding Models	46
7.1.1	CORAL	46
7.1.2	MMD	47
7.2	Results for Cross Model Knowledge Distillation in Sentence Embedding Models	48
7.2.1	Word Similarity Tasks	49
7.2.2	Sentence Evaluation Tasks	49
7.2.3	Runtime Comparison	51
8	DISCUSSION	53
8.1	Cross Model Alignment on Word Embeddings	53
8.2	Cross Model Knowledge Distillation in Sentence Embeddings	55

9 CONCLUSION	57
9.1 Summary of Findings	57
9.1.1 Cross Model Word Embedding Alignment	57
9.1.2 Cross Model Knowledge Distillation in Sentence Embeddings .	58
9.2 Future Research Directions	58
REFERENCES	60
A Appendix A	63
A.1 Algorithms	63
PUBLICATIONS FROM THE THESIS	65



LIST OF ABBREVIATIONS

BLE	Bi-Lingual Word Embeddings
CCA	Canonical Correlation Analysis
CORAL	Correlation Alignment
COS	Cosine Distance
CSLS	Cross Domain Similarity Local Scaling
EUC	Euclidean Distance
GAN	Generative Adversarial Network
ICP	Iterative Closest Point
KNN	K-Nearest Neighbor
ML	Machine Learning
MMD	Maximum Mean Discrepancy
MT	Machine Translation
MUSE	Multilingual Unsupervised and Supervised Embeddings
NLP	Natural Language Processing
OOV	Out of Vocabulary
OT	Optimal Transport
PCA	Principal Component Analysis
PCR	Point Cloud Registration
PP	Perplexity
SVD	Singular Value Decomposition
XE	Cross Entropy
XLE	Cross Lingual Word Embeddings

LIST OF FIGURES

Figure 3.1	Progressive alignment process in a 2D environment, from initial positioning to final alignment.	14
Figure 3.2	2 different data distributions (blue and red) are merged into a single distribution (green). The obtained distribution represents both distributions.	19
Figure 3.3	Illustration of point cloud registration using GMM	20
Figure 3.4	Illustration of the hubness problem in a clustered data	21
Figure 3.5	Mesh Topology, illustrating the complexity of pairwise alignments for multiple languages.	22
Figure 3.6	The evolution of ring topology from initial alignment to post-integration.	23
Figure 3.7	Star topology with a pivot language at the center, connecting to all other languages.	23
Figure 4.1	Artificially generated datasets. Left represents the same gaussian distribution. Right represents two different gaussian distributions for source and target. Blue '+' signs represent source data points where as the green crosses represent target data points.	25
Figure 4.2	The left image represents the validation set of the first dataset and its alignment to target space. The right image represents the second dataset and its alignment.	25
Figure 4.3	CORAL Domain Adaptation - S stands for Source, and T is for Target.	26
Figure 4.4	MMD with Gaussian Kernel for Domain Adaptation	27
Figure 4.5	Comparisons of word alignments within VecMap and MUSE embedding models for different language pairs.	29
Figure 4.6	Comparisons of word alignments of Glove _{en} embeddings within VecMap and MUSE embedding models for different language pairs.	29
Figure 5.1	Illustration of the distance minimization process. The two vectors represent sentence embeddings from FastText (light blue) and sBert (dark blue) for the same sentence. The goal is to minimize their angular distance, aligning FastText's sentence representations closer to those of sBert.	34

Figure 5.2	The Distance Minimization model includes a word embedding layer for refining FastText word embeddings and a transformer encoder to project the outputs. The model utilizes a combined loss function to measure the distance relative to sBert sentence vectors.	35
Figure 5.3	The Distance Perplexity Minimization model, featuring a sequence-to-sequence design with a shared embedding layer. The model utilizes FastText word embeddings and aims to preserve contextual word information, enhancing sentence perplexity for improved representations. This leads to more thoughtful reflection of knowledge in the word embedding layers.	36
Figure 7.1	Comparison of CORAL domain adaptation effects on $Glove_{en}$ and FastText (VecMap & MUSE) embeddings.	46
Figure 7.2	Left represents the MMD loss values between $Glove_{en}$ and $VecMap_{en}$, whereas the right represents the $Glove_{en}$ and $MUSE_{en}$. Blue lines indicate the training losses, red lines indicate the valiation loss values.	48

LIST OF TABLES

Table 2.1	Translations of out of vocabulary words from En $\rightarrow$ Es	6
Table 4.1	Supervised Glove to VECMAP embedding mapping results	30
Table 4.2	Supervised Glove to MUSE embedding mapping results	30
Table 6.1	Different word embedding models that are used for cross model alignment	38
Table 6.2	The validation results for <i>distance minimization</i> and <i>distance & perplexity minimization</i> methods.. The <i>distance minimization</i> approach uses combined loss of euclidean and cosine distances. Whereas the <i>distance & perplexity minimization</i> method uses combined loss of cosine and cross entropy. Each method uses equal weights between their loss functions.	45
Table 7.1	MMD loss values between source and target domains. These values are obtained without model usage.	47
Table 7.2	Results from the Google Analogy Task. sBert ₇₆₈ shows averaged sub-word results, FastText _{subset} represents the original embeddings, and FastText _{updated} includes enhancements by the Distance Min and DistancePP Min methods.	49
Table 7.3	Comparative performance of various models on classification tasks. The <i>Word Domain Coverage</i> row indicates how the classification accuracy correlates with the overlap between the training data from the WMT dataset and the vocabulary of various classification datasets.	50
Table 7.4	Results of Semantic Textual Similarity tasks on updated word embeddings and the original subset FastText word embeddings. Left values represent Pearson Correlation, whereas the right values present Spearman Correlation values.	51
Table 7.5	Execution times for sentence representation generation across different models.	51

Cross Model Alignment in Natural Language Processing

Kadir GÜNEL

Department of Computer Engineering

Doctor of Philosophy Thesis

Supervisor: Prof. Dr. Mehmet Fatih AMASYALI

In Natural Language Processing (NLP), the alignment of different model representations is a cornerstone for enhancing language understanding and processing. This study embarks on a dual-faceted exploration of cross-model alignment, specifically targeting the synchronization between diverse word embedding models and the transfer of knowledge from word to sentence embeddings.

Firstly, we dissect the alignment of word embeddings, comparing models such as GloVe and FastText, which are each trained on distinct datasets. By aligning these models, we seek to unify the linguistic information captured separately, thus enriching the representational capacity for downstream tasks. We investigate alignment techniques, specifically unsupervised methods, that can effectively bridge the gap between these models, enabling a harmonized representation of words across different training corpora.

Secondly, our research extends to the domain of knowledge transfer between models at different granularity levels—namely, from word embeddings like FastText to sentence-level embeddings such as those generated by sBERT. Here, the challenge lies in preserving the nuanced knowledge of word embeddings while scaling up to sentence-level representations. The goal is to facilitate a seamless transfer that aligns and enhances the sentence embeddings with the rich lexical semantics of word embeddings.

The convergence of these two research streams promises to not only align but also to cross-pollinate models, thus contributing to the development of NLP systems with

improved cross-lingual and cross-model versatility. This thesis proposes innovative methods for cross-model alignment and sets the stage for future research in creating more coherent, context-aware, and powerful NLP applications.

Keywords: Alignment, information distillation, information transfer, word representations, sentence representations



Doğal Dil İşlemede Modeller Arası Hizalama

Kadir GÜNEL

Bilgisayar Mühendisliği Anabilim Dalı

Doktora Tezi

Danışman: Prof. Dr. Mehmet Fatih AMASYALI

Doğal Dil İşleme (NLP) alanında, farklı model temsillerinin hizalanması, dil anlama ve işleme kapasitesini artırmanın temel taşlarından biridir. Bu çalışma, farklı kelime temsil modelleri arasında senkronizasyon ve kelime temsillerinden cümle temsillerine bilgi aktarımı özelinde çift yönlü bir çapraz model hizalama keşfine çıkmaktadır.

Öncelikle, GloVe ve FastText gibi, her biri farklı veri setlerinde eğitilmiş farklı kelime temsil modellerinin hizalanmasını detaylandırıyoruz. Bu modelleri hizalayarak farklı dilleri birleştirmeye çalışıyoruz. Böylelikle farklı görevler için temsil kapasitelerini zenginleştirmeyi hedefliyoruz. Bu modeller arasındaki boşluğu etkili bir şekilde köprüleyebilecek, özellikle denetimsiz(unsupervised) yöntemler başta olmak üzere, farklı hizalama tekniklerini kullanıyoruz.

İkinci olarak, araştırmamız bilgi aktarımı alanına uzanıyor. Daha belirgin olarak, cümle temsil kapasitesi yüksek olan bir modelden (sBERT), kelime temsil seviyesi için kullanılan ancak cümle temsil seviyesi daha zayıf olan modele doğru (FastText) bilgi aktraimini, hizalama işlemini de göze alarak, sağlamaya çalışıyoruz. Amacımız, kelime temsillerinin karşı taraftaki zengin cümle semantikleriyle hizalama yöntemiyle sorunsuz bir şekilde bilgi aktarmını sağlamaktır.

Bu iki araştırma alanının bir araya gelmesi, sadece farklı modelleri hizalamakla kalmayıp aynı zamanda modeller arası yeni melez modellerin de oluşumunu vaat etmektedir, böylece çok yönlülüğü gelişmiş NLP sistemlerinin gelişimine katkıda bulunmaktadır. Bu tez, çapraz-model hizalama için yenilikçi yöntemler önermekte ve daha tutarlı, bağlam farkındalığı yüksek ve güçlü NLP uygulamaları yaratmak için

gelecekteki arařtırmaların zeminini hazırlamaktadır.

Anahtar Kelimeler: Hizalama, bilgi damıtma, bilgi transferi, kelime temsilleri, cümle temsilleri



1

INTRODUCTION

Around the world, there are more than 7,000 different languages, yet most NLP applications are developed for only about 10 of these languages, which are predominantly anglocentric, such as English, French, and Spanish. Despite the existence of utopic text datasets, a significant portion of textual data is held by major tech companies like Google, Microsoft, Meta, Amazon, and Apple. Although these companies utilize open-source datasets, the unique internal datasets that they have, employ significantly alter the representational space of their trained models. This situation often restricts users to these companies' proprietary trained models. While knowledge distillation presents opportunities, the critical need for interoperability between different models, trained on the same or varied datasets, has not been adequately emphasized. The 'train-once, use everywhere' policy is losing its appeal, making interoperability between models increasingly vital for effective application development in both natural language processing and the broader machine learning community.

This leads us to the specific focus of our research: word embedding alignment, which involves aligning two word embedding spaces. While this problem has received extensive attention in the literature, alignment between different models remains an often overlooked aspect. After investigating the alignment process across various models and languages, our research then extends to sentence embedding models. We have developed techniques for sentence embeddings, informed by the results from word alignment model techniques. Our primary goal is to facilitate the transfer of representational information through cross-model alignment. Research in this area is scant, underscoring the originality of our work. We aim to achieve model interoperability, ensuring effective communication between models, regardless of their differences in training and architecture.

1.1 Structure of This Thesis

This research aims to enhance interoperability between various models, with a particular focus on word and sentence embedding models in diverse linguistic contexts. This study is motivated by the urgent need to improve language representation in Natural Language Processing (NLP) and to push the field of Machine Learning (ML) towards more universally adaptable models.

The structure of this thesis is outlined as follows:

- **Chapter 2** provides a comprehensive review of literature on word and sentence embeddings. This chapter is divided into two parts: the first part concentrates on word embedding models, with a special focus on alignment techniques. The second part explores sentence embedding models and their development.
- **Chapter 3** delves into key algorithmic approaches for word alignment in the NLP community, tracing their inspirations and foundations. This chapter presents essential insights into alignment models for addressing cross-model word embedding alignment that we will introduce in the next chapter.
- **Chapter 4** proposes solutions for cross-model alignment in word embedding models, introducing two distinct approaches for model calibration and their differing aspects.
- **Chapter 5** discusses methodologies for addressing cross-model alignment in sentence embedding models using knowledge distillation techniques.
- **Chapter 6** details the experimental setups employed in our research.
- **Chapter 7** presents the experimental results, divided into two sections to reflect the specific categories of our study.
- **Chapter 8** offers a separate discussion of our findings on word and sentence cross-alignment.
- **Chapter 9** concludes with our comprehensive insights on cross-model alignment, bringing together the diverse strands of our research into a cohesive summary.

1.2 Impact of This Research

Recent advancements in computing platforms have demonstrated the impressive capabilities of current hardware infrastructure, capable of processing a vast array of

data available on the internet, including text, audio, and visual formats. However, the widespread availability of this advanced hardware to end users is a process that will take time to fully materialize. In the interim, software-based outputs from various models, whether released openly or semi-openly, remain accessible and pivotal.

A prominent example is OpenAI's GPT models, renowned for their ability to provide vector representations of queries through their API. Despite their semi-open nature and often proprietary architectures, these models have made significant strides. Similarly, open-source models like Meta's LLAMA series offer accessible model weights for inference. In both scenarios, when integrating outputs from these diverse models into custom-developed models, knowledge distillation is a standard approach. However, it's typically limited to models or architectures of a similar nature.

This thesis ventures into technical approaches that are effective across a broad spectrum of models, particularly focusing on word and sentence-based models. The significance of this research extends beyond natural language processing (NLP) and permeates all major branches of machine learning (ML), where adaptability and interoperability of models are crucial.

Our research aims to set new benchmarks for model interoperability, transcending the conventional constraints of model architectures. By addressing these challenges, we seek to advance the field, unlocking the full potential of sophisticated word and sentence models for a wider range of applications. This endeavor is anticipated to greatly enhance the versatility and efficiency of ML applications, paving the way for innovative solutions across various sectors.

2 LITERATURE REVIEW

2.1 Evolution of Word Embedding Alignment

Word embeddings is the coined term for applying language model and feature learning methodologies simultaneously on the words of a given corpora. The result of this procedure generates n -dimensional vectors for each word in the corpora. These obtained vectors help computers understand the meaning of words, hence algebraic operations such as addition, subtraction, and similarities can be performed. There are various ways to obtain word vectors; one easy way is to create the co-occurrence matrix between words inside the corpus and then apply PCA for dimensionality reduction. A more popular way of obtaining word vectors is to use neural network optimization (Mikolov, Sutskever, et al. 2013), (Pennington, Socher, Manning 2014). Initially, each word in the corpus is initialized with a random n -dimensional vector. Each vector is processed according to its neighboring words, hence conceptual properties are obtained at the end of the optimization procedure. All applications related to NLP and ML are built on top of word vectors, which makes them indispensable.

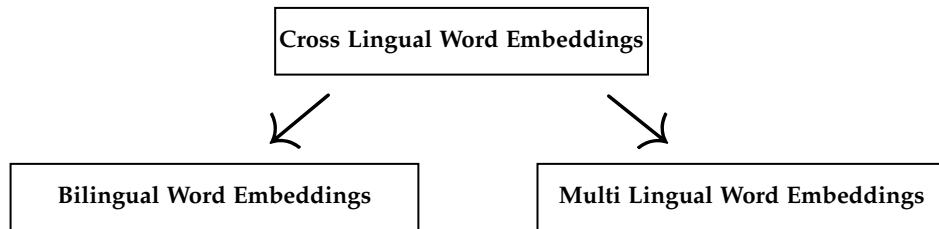


Figure 2.1 Taxonomy of Cross-lingual Word Embeddings

Cross-lingual word embeddings is the methodology for mapping one embedding space (source space) to another (target space). The aim is to find the proper word translation of each word inside the embedding space from one language (L_s) to

another (L_t). This procedure can be computed using supervised or unsupervised machine learning techniques. Conceptually, to map monolingual data (L_s) to target space (L_t), word embedding matrices of both language pairs are needed, and for the mapping operation, a special matrix, called the translation matrix, is used.

The first approaches to cross-lingual embeddings were done between two language pairs, but recently the mapping procedure has been generalized to multiple language pairs. Hence, in literature, cross-lingual word embeddings are divided into two subcategories. The term Bilingual word embeddings is used for the mapping procedure between two language pairs, and the term Multilingual word embeddings (or Cross Lingual Embeddings) is used for the mapping procedure between more than two language pairs (see Figure 2.1).

From the methodological perspective, cross-lingual word embeddings can again be divided into two subcategories: supervised and unsupervised approaches. The supervised approaches require the use of pre-aligned vocabularies, known as bilingual vocabularies. Methods that use bilingual vocabularies can be realized by using iterative methods and analytical methods. The other category does not necessarily use pre-aligned vocabularies between language pairs but tries to build one by using a small initial vocabulary (for example, 25 words).

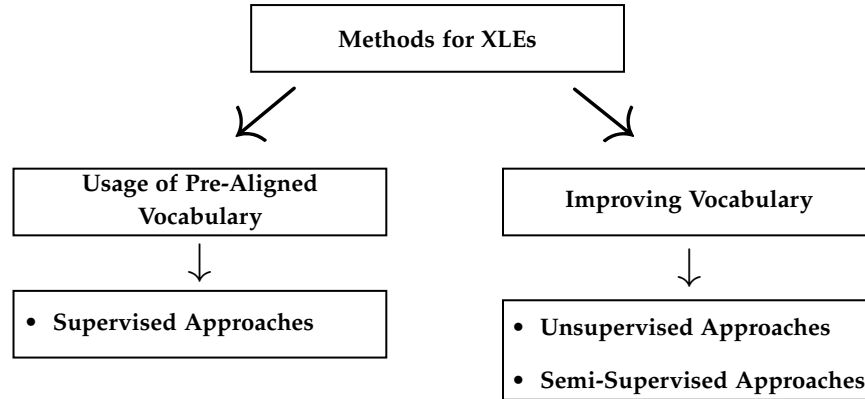


Figure 2.2 Methodologies for obtaining Cross-lingual Word Embeddings

2.1.1 Bilingual Word Embeddings

The study (Mikolov, Quoc V Le, Sutskever 2013) is one of the first to focus on bilingual word embeddings. Their research observed that words in the parallel corpora of different languages could have linear relations when reducing the dimensionality of word vectors. This empirical finding is leveraged to mitigate the out-of-vocabulary problem in machine translation systems.

For their experiments, English-Spanish parallel datasets from WMT were utilized. word2vec model was used to obtain word embedding spaces for both English and Spanish. Depending on the translation direction, the most frequent 5,000 words were selected from the English corpora for training, and the subsequent 1,000 frequent words were chosen for testing. The Spanish translations of the training and testing datasets were obtained using the Google Translate service available at that time.

The model employed for the mapping operation is least squares regression (refer to equation 2.1), where x_i and z_i represent the word vectors of the source and target languages, respectively. n denotes the dimension size of the word vectors, and W , termed the "Translation Matrix," is the method's core parameter.

$$\min_W \sum_{i=1}^n \|Wx_i - z_i\|^2 \quad (2.1)$$

Equation 2.1 is trained using the gradient descent algorithm. During testing, any source word ($word_s$) is multiplied by the translation matrix W to obtain a vector. This vector represents a concept to be located within the target embedding space, for which the well-known knn algorithm with a cosine similarity metric is employed. The results for En $\rightarrow$ Es translation are shown in table 2.1.

Table 2.1 Translations of out of vocabulary words from En $\rightarrow$ Es

English Word	Computed Spanish Translations	Dictionary Entry
pets	mascotas	mascotas
mines	minas	minas
unacceptable o	inaceptable	inaceptable
prayers	oraciones	rezos
shortstop	shortstop	campocorto
interaction	interaccion	interaccion
ultra	ultra	muy
beneficial	beneficioso	beneficioso
beds	camas	camas
connectivity	conectividad	conectividad
transform	transformar	transformar
motivation	motivacion	motivacion

The above approach uses a gradient descent solution (iterative approach) to find the translation matrix W , which is based on heuristics. (Faruqui, Dyer 2014) developed an analytical solution for obtaining bilingual word embeddings, using the Canonical Correlation Analysis. Their approach aims to reflect linear projections from source to target language by maximizing the correlation between source and target embeddings, claiming an improvement in the quality of monolingual embeddings.

(Artetxe, Labaka, Agirre 2018b) proposes a multi-step framework for generalizing the mapping of embedding spaces, which includes steps like normalization, whitening, orthogonal mapping, re-weighting, de-weighting, and dimensionality reduction. Their iterative process manipulates source embeddings using linear algebra methods, showing superior performance over previous works for various language pairs.

(Artetxe, Labaka, Agirre 2016) translates the least squares regression model into a closed-form solution using linear algebra. Following this, (Artetxe, Labaka, Agirre 2017) developed a "weakly supervised" approach by reducing the vocabulary size of the pre-aligned bilingual dictionary from 5,000 words to 25. This approach uses an iterative method for obtaining bilingual word embeddings.

$$W^* = \arg \min_W \sum_i \sum_j \mathcal{D}_{ij} \|X_{i*}W - Z_{j*}\|^2 \quad (2.2)$$

In equation 2.2, X and Z represent the embedding matrices of the source and target languages. X_{i*} and Z_{j*} are the i th and j th vector representations of the source and target languages, respectively. A binary matrix $\mathcal{D}$ signifies alignment between the i th and j th words, dropping the assumption of pre-aligned vocabulary. $\mathcal{D}_{ij} = 1$ if the i th and j th words are aligned, and 0 otherwise. The goal is to learn a translation matrix W that minimizes the Euclidean distance within $\mathcal{D}_{ij}$.

In their groundbreaking study, (Artetxe, Labaka, Agirre 2018a) introduced a novel method for initializing the seed dictionary D , paving the way for fully unsupervised learning of bilingual word embeddings. This innovative approach involves using the most n frequent words from both the source and target languages to construct a seed dictionary without supervision, representing a significant advancement in the field of natural language processing.

(Heyman et al. 2019) describes an unsupervised methodology for learning bilingual word embeddings, dividing the process into two steps: Mapping and Refinement. In mapping, the translation matrix W is calculated from source to target language using various algorithms like regression, CCA, etc. The refinement step involves processes like whitening, denoising, re-weighting, and dimensionality reduction to improve the aligned embedding spaces of source and target, enabling better alignments in subsequent iterations.

The main advantage of the unsupervised approach over supervised methods is that it does not require pre-aligned bilingual vocabularies, opening possibilities for low-resource languages.

2.1.2 Multilingual Word Embeddings

The pioneering approach to adapting cross-language word vectors for multiple languages was developed by (Ammar et al. 2016). Their study encompassed over 50 languages. They found that word vectors, formed by conjugating these languages into a common target language (English), outperformed those created solely for the target language.

The methods employed in their study are multi-cluster and multiCCA. Both approaches require only a bilingual dictionary and monolingual data for each language pair, eliminating the need for a parallel corpus. In the multi-cluster method, translations are grouped by their concepts using dual language dictionaries, forming a unified language corpus. This approach ensures that word representations within each cluster are identical. The resulting corpus is then trained using the skip-gram method, with words replaced by their translations within the clusters. The MultiCCA method significantly extends the Canonical Correlation Analysis method, initially used by (Faruqui, Dyer 2014), to obtain multiple word vector representations for the target language, English.

The resulting multilingual word vectors were evaluated in two ways: intrinsic and extrinsic. For intrinsic evaluation, methods like word similarities, word translations, and relational assessments were used. Extrinsic evaluation involved testing the multilingual word vectors on NLP tasks such as classification and parsing. Their results show better word representations compared to monolingual techniques.

Another approach to deriving multilingual word vectors was introduced by (Heyman et al. 2019). This method involves incrementally adding new languages to an existing multilingual word space, thus enhancing the interconnections between languages. The paper proposes two approximations. The first is synchronizing all languages to a single common language, termed Single Hub Space, similar to the previously mentioned study. The second, Incremental Hub Space, involves mapping a new language (L_3) onto a previously established language pair (L_1, L_2). This method demonstrated superior performance, especially between linguistically distant language pairs, and yielded reliable multilingual word vectors. The vectors obtained were applied to classification and parsing tasks, showing improved results over the previous study.

2.2 Sentence Embedding Techniques

The development of sentence vector representations has been a prolific area of study. Initial methods focused on averaging pretrained word embeddings, a

technique showcased in studies by (Mikolov, K. Chen, et al. 2013), (Pennington, Socher, Manning 2014) (Quoc V. Le, Mikolov 2014). Building on this foundation, (Bojanowski, Joulin 2017) expanded representation capabilities from individual words to variable-length texts, including sentences, paragraphs, and documents. This approach empirically surpassed the traditional bag-of-words model in various text classification and sentiment analysis tasks, overcoming limitations inherent in the conventional averaging method. Subsequent developments involved techniques like subtracting principal components (Arora, Liang, Ma 2017) and employing the skip-gram model, as presented by (Kiros et al. 2015), which (Logeswaran, Lee 2018) later optimized for faster training.

Concurrently, advancements in neural architectures, notably the Attention mechanism introduced by (Vaswani et al. 2017), have revolutionized sentence processing. A variety of models, such as those by (Cer et al. 2018), (Radford, Narasimhan 2018) have emerged, balancing accuracy and computational efficiency and excelling in diverse language tasks. The emergence of sophisticated models like BERT (Devlin et al. 2019, pp.) RoBERTa (Yinhan Liu et al. 2019), and ALBERT (Lan et al. 2020) has expanded the scope of transfer learning and catalyzed progress in knowledge distillation within language understanding tasks. Researchers have explored various sentence embedding strategies, with a notable trend being the use of siamese network architectures that utilize BERT’s capabilities. This innovative approach, known as sBERT, leverages BERT’s advanced deep learning features to efficiently generate embeddings that capture complex semantic relationships within sentences (Reimers, Gurevych 2019).

Emerging from this context, knowledge distillation as a concept has been rigorously explored as a strategic means to transfer the essence of complex models into more simplified counterparts (Hinton, Vinyals, Dean 2015). Typically, these approaches involve models with identical architectures, utilizing logits (pre-softmax outputs) from a teacher model to guide the student model towards producing similar outputs. (Mao, Nakagawa 2023) and (Sahlgren 2021) have integrated these approaches for language-agnostic sentence embeddings. Notably, these methods do not attempt to transfer knowledge between different architectures.

As large-scale pre-trained models become more common in NLP, managing them within limited computational resources is increasingly challenging. In response, (Sanh et al. 2020) introduced DistilBERT, a smaller language representation model that applies knowledge distillation during pre-training. This model is designed for practical applications requiring robust language comprehension but limited computational resources.

Despite the complexity of neural networks like BERT and GPT, simpler neural networks can still compete if enhanced through knowledge distillation from these advanced models. (Tang et al. 2019) showed that a single-layer BiLSTM, and its siamese variant, can match the performance of more complex models with fewer parameters and reduced inference time, thanks to distillation.

In a novel approach, (Yufan Liu et al. 2022) demonstrated knowledge transfer from a transformer model to a convolutional model in image classification. This method aligns student and teacher features in two projected feature spaces.

Our study introduces an innovative approach in sentence embeddings and knowledge transfer, employing a unique distillation method. Unlike previous works, our method integrates the distinct architectures of pretrained FastText word embeddings with sBERT sentence vector representations. This bridges the gap between robust sentence representation models and the need for computationally feasible solutions, setting a new direction for future research in the field of cross model alignment.

Before diving into cross model word alignment, in this chapter, we first introduce the word embedding alignment procedures in literature and which algorithm(s) are placed under them. For this purpose we introduce the Point Cloud Registration. This is a well studied area in computer vision. Then we introduce the Iterative Closest Point algorithm which is very well known in computer vision. Then make relate it with the bilingual and cross lingual embedding methods. We also give details about other methods that are being used in natural language processing community and other computer science areas. The alignment process is mainly about finding appropriate neighbors despite the capabilities of used models. Hence we give information about the searching strategies in alignment. Finally, we give topological overview of alignment.

Furthermore, we explore various other methods employed in the natural language processing community and broader computer science fields. The core of the alignment process revolves around finding suitable neighbors, regardless of the models' capabilities. Therefore, we also discuss different search strategies used in alignment. An overview of the topology involved in alignment is provided, highlighting its significance in this context.

Although our experiments in this thesis do not directly concentrate on cross word embedding alignments, we believe it is pertinent to delve into the details of cross-model alignment for word embedding models. While our experimentation was limited to using star topology for cross-model alignment of word embeddings, the strategies outlined here could serve as a source of inspiration for future research endeavors in this area.

In our quest to better understand alignment processes, we have ported the source code of vecmap, as proposed by (Artetxe, Labaka, Agirre 2018a) and originally written in Python, to Julia. Additionally, we implemented the Optimal Transport approach as suggested by (Grave, Joulin, Berthet 2018).

3.1 Point Cloud Registration

Point Cloud Registration (PCR) is a crucial and well-established problem in the field of Computer Vision. The goal of PCR is to align different datasets by identifying and matching the most relevant or closest data points between them. Interestingly, this alignment process finds a parallel in Natural Language Processing (NLP), where similar techniques are applied to align word embedding spaces across various languages.

In this section, we delve into the concept of PCR, exploring its interpretation from two distinct perspectives that aid in aligning two or more point clouds. We examine methods from both NLP and Computer Vision, acknowledging that NLP often draws inspiration from the latter. The cross-pollination of ideas between these fields suggests the potential for models that are applicable to both domains.

At its essence, Point Cloud Registration involves aligning two or more point clouds in such a way that corresponding points across these clouds demonstrate a high level of similarity. Typically, a PCR algorithm undertakes two key tasks:

1. Identifying relationships between points, which requires an effective search algorithm.
2. Implementing a model to adjust one point cloud to align accurately with another.

Within this thesis, we recognize that Bilingual Embeddings (BLEs) and Cross-lingual Embeddings (XLEs) are specific applications of the broader PCR problem. Our focus, however, will be predominantly on cross model alignment discarding the distinction between BLEs and XLEs.

The alignment process in PCR can be broadly categorized into two independent perspectives:

1. **Mathematical view of approaches:** This perspective covers the theoretical and computational frameworks that underpin the models.
2. **Topological of Approaches:** This aspect involves examining the structural and relational characteristics of the models.

Before exploring the various alignment approaches, it is essential to understand the Iterative Closest Point (ICP) algorithm. ICP serves as the foundational basis for the 'vecmap' framework approach employed in this research.

3.1.1 Iterative Closest Point

For aligning two distinct word embedding spaces, our primary focus centers on a specific framework known as vecmap¹. Vecmap is versatile, employing a range of machine learning approaches including supervised, semi-supervised, and unsupervised learning for word embedding alignment. Central to this framework is the computation of a rotation matrix, designated as W , using the Singular Value Decomposition (SVD) method. This computation entails multiplying the source space (X) by the target space (Y).

This SVD-based approach, established as optimal in the study by (Besl, McKay 1992), is a fundamental component of the Iterative Closest Point (ICP) algorithm. Similar to Point Cloud Registration (PCR) in computer vision, unsupervised bilingual word representation mapping is essentially an alignment challenge. However, a significant difference lies in the dimensionality of the data points involved. While PCR typically deals with 3-dimensional data points, the dimensions in word embedding spaces can be much more extensive, reaching up to 300 dimensions or more.

The ICP algorithm, and by extension vecmap, employs various strategies for alignment. These include:

1. **Point Subsets:** This can range from using all points to uniform or random sampling and feature-based approaches.
2. **Data Association Strategies:** These vary from the closest point, closest compatible point, to point-to-plane strategies.
3. **Weight Correspondences:** Important for the alignment accuracy.
4. **Rejecting Potential Outlier Point Sets:** Given that both ICP and vecmap are based on linear regression models, outliers can significantly affect their performance.

Figure 3.1 show the progressive alignment of the ICP algorithm in two different point clouds. The upper left-most image presents two data point clouds in a 2D environment, with red dots symbolizing true values and blue dots representing data to be aligned. The image to the right depicts the preprocessing step, where data clouds are centered in space. The upper right-most image illustrates the initial guess. The bottom left image captures the iterative alignment process, mapping the most relevant data points to each other. The final image in the bottom right displays the completed

¹<https://github.com/artetxem/vecmap/tree/master>

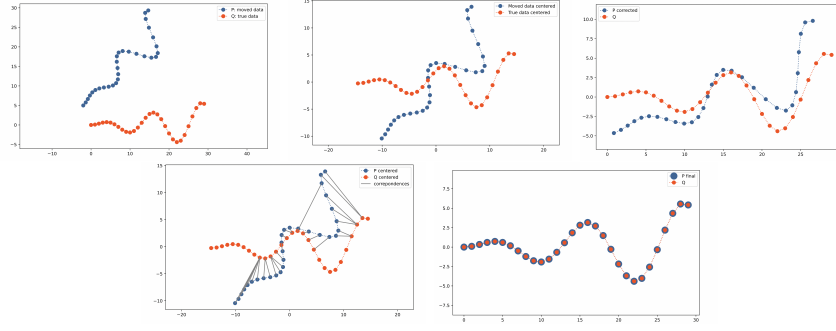


Figure 3.1 Progressive alignment process in a 2D environment, from initial positioning to final alignment.

alignment². *Vecmap* uniquely tackles the alignment problem without any supervision. It initially considers the first n words from the embedding space for both the source and target.

3.1.2 Least Squares & Orthogonal Procrustes Problem

The least squares problem, a fundamental concept in optimization, can be addressed using various techniques. One notable method is known as Orthogonal Procrustes. In this subsection, we will transform the least squares equation (referenced as eq. 3.1) into an orthogonal Procrustes problem.

$$\min_W \|WX - Y\|^2 \quad (3.1)$$

To convert the least squares problem into an orthogonal Procrustes problem, an orthogonality constraint is added. The modified equation, incorporating this constraint, is presented below:

$$\text{s.t. } W^T W = I \quad (3.2)$$

The process of finding the nearest orthogonal matrix (W) involves calculating the product (YX^T) and subjecting it to singular value decomposition (SVD). This decomposition yields matrices (U), (S), and (V^T). Subsequently, the orthogonal matrix (W) is derived by multiplying (UV^T).

$$U, S, V^T = \text{svd}(YX^T) \quad (3.3)$$

²The set of images are taken from <https://nbviewer.org/github/niosus/notebooks/blob/master/icp.ipynb>

$$W = UV^T \quad (3.4)$$

The transformation of the least squares problem into an orthogonal Procrustes problem allows for a more constrained and potentially more accurate solution, especially in contexts where the orthogonality of W is a desirable property. All bilingual alignment tools use the orthogonal approach for finding proper candidates between source and target languages.

3.2 ICP for Unsupervised Word Embedding Alignment: *vecmap*

The methodology developed by (Artetxe, Labaka, Agirre 2018a) in their robust approach to word embedding alignment can be delineated into five primary steps:

1. **Preprocessing:** Initially, both input spaces undergo unit and center normalization. This standardization ensures that words within both embedding spaces are of equal size and centrally aligned. Such normalization is crucial for maintaining consistency and balance in the embedding spaces.
2. **Initialization:** This crucial step involves the extraction of a seed dictionary by finding correlations between subsets of source (X) and target (Y) word embeddings. The aim is to establish an initial mapping, forming a foundational basis for further refinement and alignment processes.
3. **Training:** The core of the method is the iterative determination of a translation matrix W . This matrix is refined by utilizing the correlations identified between corresponding words in the source (X) and target (Y) embeddings. Singular Value Decomposition (SVD) is employed in this step to ensure the matrix W accurately captures the relationship between the two embedding spaces.
4. **Saving the Mappings:** Upon reaching the algorithm's stopping condition — a predefined criterion indicating the alignment process's completion — the aligned source (X) and target (Y) embeddings are saved. This step finalizes the alignment, preparing the embeddings for use in subsequent tasks or analyses.
5. **Post-Processing (Optional):** As an optional but often beneficial step, post-processing techniques such as whitening are applied to both the source and target spaces. These techniques can further refine and enhance the quality of the aligned embeddings, particularly in applications requiring a high degree of precision or specificity.

The first, fourth and the final steps are about the normalization of the embedding spaces for better neighborhood search and saving the aligned spaces.

We will explain the initialization and the training steps in depth.

3.2.1 Initialization

The algorithm operates without explicit supervision at any stage. Initially, it faces the challenge of aligning source and target words. To address this, the authors employ an innovative, albeit counterintuitive, solution. They explore whether correlations (similarity matrices) of the first 'n' words in each embedding space can help build a seed vocabulary, even if these words are seemingly unrelated. Surprisingly, this method proves effective in constructing a dictionary for initialization. The process requires four correlation matrices:

1. A matrix for the source, X_{sim} ,
2. A matrix for the target, Y_{sim} ,
3. A transition matrix from X_{sim} to Y_{sim} , denoted as Sim ,
4. A transition matrix from Y_{sim} to X_{sim} , denoted as Sim^T .

The latter two matrices are derived using the first two. Notably, before computing the Sim matrix, values in the correlation matrices X_{sim} and Y_{sim} are sorted. This sorting step has been found to aid the algorithm's convergence, requiring fewer iterations compared to its non-sorted counterpart. Additionally, observing the singular values of both X_{sim} and Y_{sim} reveals that sorting regularizes the energy levels of the resulting Sim matrix.

Both similarity matrices are then processed using the Cross-domain Similarity Local Scaling (CSLS) algorithm (Conneau, Lample, et al. 2017), which identifies the top k similar word indices for both directions ($X \rightarrow Y$ and vice versa). This step yields the most appropriately similar word indices for both sides see Algorithm 2 for algorithmic description.

3.2.2 Training

The objective of the training phase is to iteratively find a new set of candidate words that align more accurately than in the previous iteration. This is achieved through the following steps at each iteration:

1. **Calculating Similarities:** The algorithm calculates the similarities between the source and target embeddings. This process results in a similarity matrix which captures the relationships between words in the source and target languages.
2. **Applying Singular Value Decomposition (SVD):** The similarity matrix is then subjected to SVD, from which the orthogonal matrices U and V are derived. These matrices are crucial as they satisfy the orthogonality condition of the problem, a key requirement for accurate alignment.
3. **Computing the Translation Matrix W :** The orthogonal matrices U and V facilitate the computation of the translation matrix W . This matrix is instrumental in transforming the source word embeddings.
4. **Rotation of Embeddings:** The original source word embeddings X are multiplied by the translation matrix W , resulting in 'rotated embeddings'. This rotation aligns the source embeddings more closely with the target embeddings.
5. **Finding New Candidate Words:** A new set of candidate word indices is identified using the Cross-domain Similarity Local Scaling (CSLS) algorithm. This algorithm operates in both directions (source to target and vice versa), ensuring a comprehensive alignment.
6. **Iterative Process:** This entire process is repeated iteratively until the algorithm's objective function reaches a predefined threshold value ξ .

For a detailed overview of the algorithm, please refer to 3 in Appendix~A.1.

3.3 Other Metods for Alignment

The process of aligning word embeddings can be conceptualized as rotating one point cloud towards another. Previously we have primarily focused on how the vecmap algorithm manages this alignment using the Procrustes method. However, there are three additional methodologies exist for the alignment process. First two of these approaches are used directly from natural language processing community. The third approach, originating from the field of computer vision, is geared towards aligning different distributions, a concept that can be analogously applied in NLP scenarios.

The first methodology pertains to Bilingual Word Embeddings and employs Optimal Transport (OT), as discussed in (Grave, Joulin, Berthet 2018). This approach leverages the principles of OT to effectively align word embeddings across different languages.

The second approach proposed by (Conneau, Lample, et al. 2017) uses Generative Adversarial Networks (GANs) for alignment. This approach uses dual-objective strategy for discrimination and mapping between source and target words.

The third approach, derived from the field of computer vision, concentrates on multiple point cloud registration. This methodology, explored in (Evangelidis, Horaud 2017), applies techniques from computer vision to address the challenges associated with aligning multiple point clouds, a concept that can be analogously applied to the alignment of word embedding spaces.

These diverse methodologies demonstrate the evolving landscape of word embedding alignment techniques, each contributing unique solutions to the challenges in this field.

3.3.1 Optimal Transport

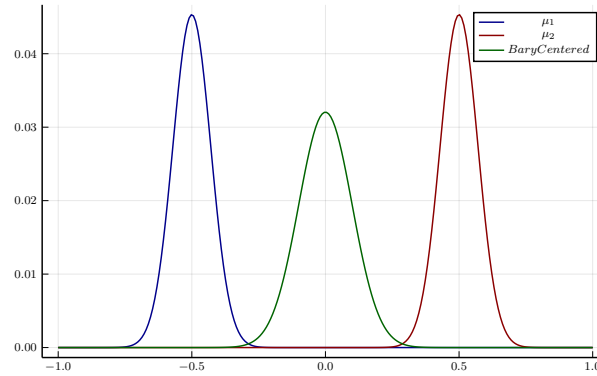
Optimal Transport aims to solve minimum cost flow problem. The basic idea is to map one distribution towards another which is interpreted as an alignment process by (Alvarez-Melis, Jaakkola 2018), (Grave, Joulin, Berthet 2018) for Bilingual Word Embeddings. The only difference between both papers is the initialization step where the first guess between source and target words are made. The rest of both papers use sinkhorn algorithm with entropic regularization for optimal transport and the rotation matrix is calculated by Procrustes; just like in vecmap. In addition, there are 2 more papers that are worth noticing. Both of these papers propose to use Optimal Transport for obtaining XLEs: (Alaux et al. 2018), (Lian et al. 2020).

(Alaux et al. 2018) uses pivoting (star topology) where as (Lian et al. 2020) uses a total different approach which it does not require any pivoting: Wasserstein Barrycenter. Wasserstein Barrycenter finds a common distribution between n distributions which can represent all different data sets (see figure 3.2) .

There are three critical distinctions between the vecmap algorithm and the OT approach as described by (Grave, Joulin, Berthet 2018):

1. **Alignment Flexibility:** In vecmap, a word from either the source or target side can be aligned with one or more corresponding words, offering a degree of flexibility in the alignment process.
2. **Word Selection Method:** While vecmap consistently utilizes the first 20,000 words in each iteration for alignment, (Grave, Joulin, Berthet 2018) 's OT

Figure 3.2 2 different data distributions (blue and red) are merged into a single distribution (green). The obtained distribution represents both distributions.



method employs a different strategy by selecting words randomly from the first 20,000.

3. Post-Processing: The OT approach, unlike vecmap, does not incorporate any post-processing steps in its alignment process.

We also ported code of (Grave, Joulin, Berthet 2018) to Julia which its original implementation can be also found under FastText repository in Github. The results are comparable to *vecmap* results for mappings from English to German, Italian, Finnish and Spanish but not better than *vecmap*.

3.3.2 Generative Adversarial Networks

(Conneau, Lample, et al. 2017) propose the use of a generative adversarial neural network as a domain adversarial tool for the alignment operation. The network employs two main objectives during training: discrimination and mapping. The discriminator component is tasked with making accurate distinctions between randomly rotated samples from the source and target domains. In contrast, the mapping component is responsible for adjusting the embedding space based on feedback from the discriminator network. This dual-objective approach enables the model to effectively align source and target words.

3.3.3 Gaussian Mixture Model

(Evangelidis, Horaud 2017) propose an alignment method based on the assumption that point clouds (interpreted as embedding spaces in NLP) are normally distributed. This method involves internally segmenting each cloud and then aggregating elements similar to those in other clouds. This aggregation leads to the creation of a

‘universal embedding space’, achieved by using the mean and variance from different data clouds. The alignment process is iteratively refined using the Expectation Maximization algorithm, specifically the Expectation Conditional Maximization technique, as cited in (Meng, Rubin 1993). Their approach shows improved performance compared to the Iterative Closest Point algorithm.

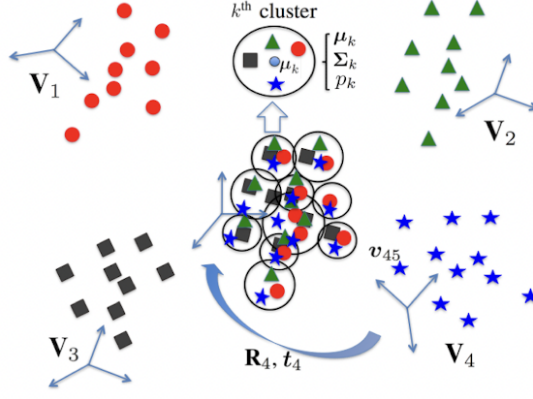


Figure 3.3 Illustration of point cloud registration using GMM

In order to describe their approach for multiple cloud registration see figure 3.3 - image directly taken from the original paper. There are V_i clouds, an observed point, e.g. $V_{45} \in V_4$, once rotated and translated from the set-centered coordinate frame to the mixture-centered coordinate frame (R_4 and t_4) is assigned to the k_{th} mixture component defined by μ_k , Σ_k and p_k . p_k is the mixing coefficient where $\sum_{k=1}^K p_k = 1$

3.4 Similarity Searching: *KNN* vs. *CSLS*

In word alignment tasks, where capturing the semantics of words is crucial, choosing the right search method is key. Cosine similarity, often used with the k-nearest neighbor (knn) search method, is a commonly employed metric for this purpose. However, knn faces a significant challenge known as the hubness problem.

To understand the hubness problem, consider a 2-dimensional space filled with many data points, most clustered in a specific area, with an outlier far from this cluster (Figure 3.4). When applying the knn algorithm to this dataset, the algorithm consistently identifies the nearest neighbors from the clustered points for the outlier. This reciprocity is not observed for points within the cluster, leading to asymmetric nearest neighbor relationships and the hubness issue.

To address this, (Conneau, Lample, et al. 2017) proposed the Cross Domain Similarity Local Scaling (CSLS) method. CSLS mitigates the hubness problem by penalizing the identified vectors based on the average similarity of each point to its neighbors.

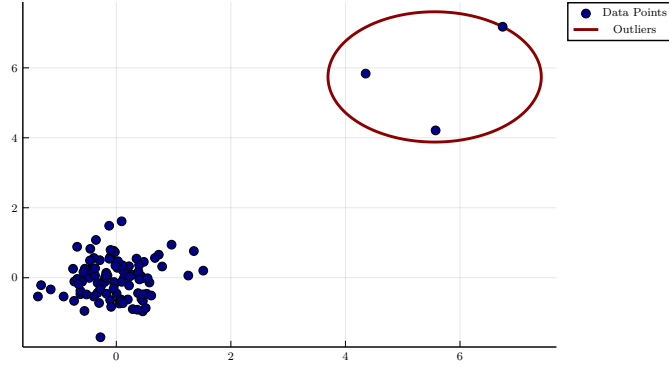


Figure 3.4 Illustration of the hubness problem in a clustered data

The core calculation of CSLS, as shown in Equation 3.5, involves the translation matrix W , a source word x_s , and a target word y_t , where $\cos$ represents the cosine similarity between x_s and y_t . The terms $r_T(Wx_s)$ and $r_S(y_t)$ denote the mean similarity of x_s to its target neighborhood and y_t to its neighborhood, respectively.

$$CSLS(Wx_s, y_t) = 2 \times \cos(Wx_s, y_t) - r_T(Wx_s) - r_S(y_t) \quad (3.5)$$

In Equation 3.5, the cosine similarity is effectively doubled, and the mean similarities from both the source to the target space and the target to its neighbors are subtracted as penalization terms.

$$r_T(Wx_s) = \frac{1}{K} \sum_{y \in N_T(Wx_s)} \cos(Wx_s, y_t) \quad (3.6)$$

Empirical studies by (Conneau, Lample, et al. 2017) show that CSLS provides more accurate neighbor identification compared to the traditional knn approach.

3.5 Topological View of Alignment Models

Graph theory plays a crucial role in elucidating the topology of alignment processes within cross-lingual embeddings (XLEs). By employing graph-theoretical principles, we gain the ability to systematically organize and interpret the complex interconnections among multiple languages in XLE systems. This section is dedicated to examining three primary topologies, each offering unique insights and strategies for addressing the alignment challenge in XLEs.

Each topology presents its distinct advantages and challenges. The mesh topology, although comprehensive in its coverage, demands significant computational resources.

In contrast, the ring topology, with its streamlined structure, provides a more practical solution, particularly well-suited for sequential or pipeline processing. The star topology, centered around a pivot language, strikes a balance, making it especially beneficial in scenarios where a common reference language is applicable.

In conclusion, the selection of an appropriate topology in XLE systems is contingent upon several factors, including the specific linguistic task at hand, the number of languages involved, and the available computational resources. A thorough understanding of these topologies through the lens of graph theory not only facilitates the development of more efficient and effective XLE systems but also enhances our grasp of the complex relationships between diverse languages.

3.5.1 Mesh Topology

In a mesh topology, each language is aligned with every other language individually. This comprehensive approach, though thorough, faces significant scalability challenges. For (n) languages, it requires training $(k = \frac{(n-1)n}{2})$ different models. While this ensures exhaustive pairwise alignment, the quadratic growth in model training makes it impractical for scenarios involving multiple languages. Consequently, mesh topology is rarely feasible for large-scale XLE systems due to its computational and resource-intensive nature (Figure 3.5).

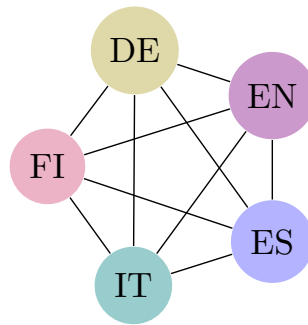


Figure 3.5 Mesh Topology, illustrating the complexity of pairwise alignments for multiple languages.

3.5.2 Ring Topology

As a more scalable alternative, ring topology simplifies the alignment process. Here, languages are arranged in a sequential manner, forming a ring-like structure (Figure 3.6). Each language aligns only with its immediate neighbors in the sequence, eventually circling back to the original language. This topology requires just (n) connections for (n) languages, significantly reducing complexity compared to the mesh topology. The alignment process in ring topology forms a closed loop.

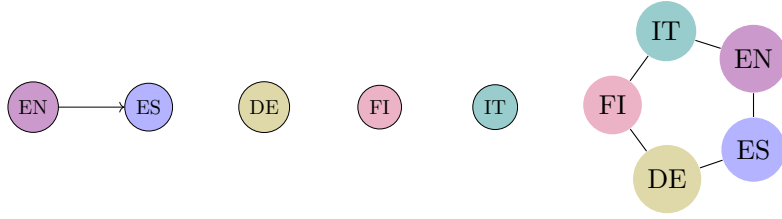


Figure 3.6 The evolution of ring topology from initial alignment to post-integration.

3.5.3 Star Topology

In the star topology, a central pivot language is chosen, typically a widely spoken language like English, around which all other languages are aligned (Figure 3.7). This topology is exemplified in FastText’s unsupervised multi-lingual alignment model as described by (Grave, Joulin, Berthet 2018). The pivot language serves as a universal reference, simplifying the alignment process by reducing it to multiple one-to-one mappings between the pivot and each of the other languages.

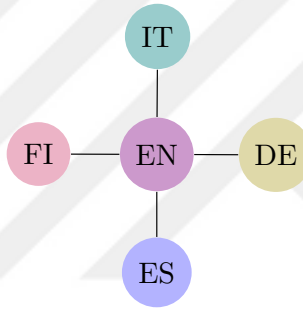


Figure 3.7 Star topology with a pivot language at the center, connecting to all other languages.

CROSS MODEL WORD EMBEDDING ALIGNMENT

In this thesis, we address the challenges posed by the use of diverse data sets and model hyperparameters by different research groups for generating different word embeddings models. These variations can lead to embeddings spaces with differing distributions. Such disparities often result in misalignments when attempting to align embeddings in an unsupervised manner. To counter this issue and maintain the unsupervised nature of the alignment process, this thesis focuses on unsupervised domain adaptation methods. By doing so, we aim to control the data distribution shift between datasets used by different research groups. This approach is intended to maximize the generalization of the alignment models being developed. Additionally, by minimizing reliance on any supervised methods, we strive to utilize the available data more effectively, enhancing the overall efficiency and applicability of our models.

4.1 Overview of Problem

We observe how the unsupervised algorithm behaves in controlled data sets. Hence dummy data sets are generated. Then we observe how each step of the algorithm changes its inputs, and their effects to the outcome.

In order to visualize how the algorithm works between similar and different data sets, two artificial data sets were created. The first artificial dataset is created using a single gaussian and its counterpart (target side) is created by using a 2 by 2 rotation matrix (R) with θ being 60 degrees. The second artificial dataset is generated by using two different Gaussian distributions (figure 4.1).

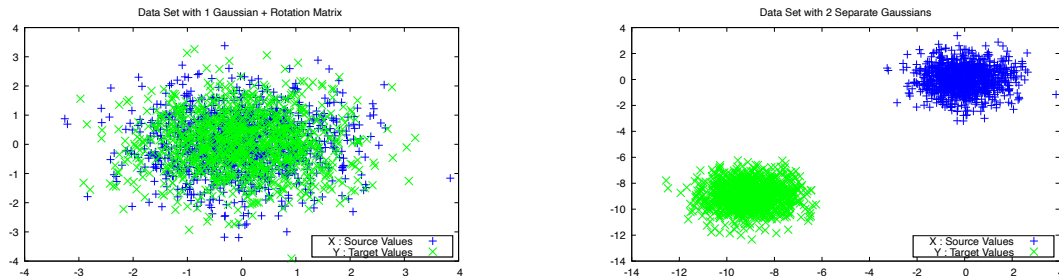


Figure 4.1 Artificially generated datasets. Left represents the same gaussian distribution. Right represents two different gaussian distributions for source and target. Blue '+' signs represent source data points where as the green crosses represent target data points.

Since the target data points in the first dataset is generated directly from the source data points, the unsupervised algorithm aligns them close to perfect. On the other hand, the second dataset where source and target data points are sampled from two different gaussian distributions where the unsupervised algorithm cannot align them at all (figure 4.2).

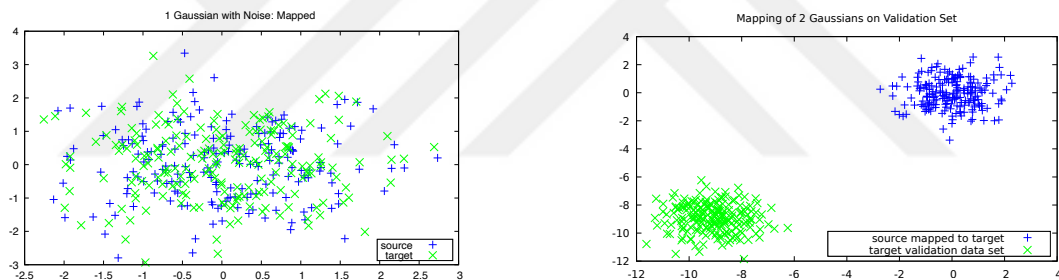


Figure 4.2 The left image represents the validation set of the first dataset and its alignment to target space. The right image represents the second dataset and its alignment.

The second dataset is used to show the reader that when the data distributions are different from each other, the unsupervised alignment algorithm does not work as expected. In order to alleviate the distributional differences between different word embedding models, it is intended to explore unsupervised *domain adaptation* techniques.

4.2 Domain Adaptation Techniques for Cross Model Alignment

In our endeavor to align word embeddings across different models like GloVe and FastText, we faced significant challenges with the method as implemented in the VecMap approach. These challenges were more pronounced when aligning models

that originated from different frameworks, as opposed to those produced within the same environment. This discrepancy became especially evident in our experiments, where the results were not comparable. This issue was further compounded when attempting to align embeddings from different languages, which were independently generated by various research laboratories. In many cases, our alignment efforts either led to sub-optimal results or even no outcomes for most of the time.

To effectively address these challenges and facilitate a successful analysis and alignment of word embeddings from diverse sources and models, we turned to two distinct adaptation techniques: CORAL and MMD.

The key distinction between these two approaches lies in their focus and methodology. CORAL primarily operates on the feature spaces of the word embeddings. It aims to align these spaces by minimizing domain shift, focusing on the second-order statistics of the distributions. On the other hand, MMD takes a different approach by calculating domain differences. It considers the sample space and employs a statistical measure to gauge the disparity between the source and target domain distributions. This fundamental difference in their operational focus – feature spaces for CORAL and sample spaces for MMD – is central to understanding their unique roles in our word embedding alignment process.

4.2.1 Correlation Alignment (CORAL)

CORAL(Sun, Feng, Saenko 2016) is designed to minimize domain shift between different datasets. It achieves this by aligning the second-order statistics (covariances) of source and target domain features. In simpler terms, CORAL adjusts the data distributions of different datasets (or domains) so that they become more similar to each other.

CORAL is primarily an unsupervised technique. It does not require labeled data from the target domain, which makes it crucial for our scenario where different word embedding models in different languages need to be aligned in an unsupervised manner.

```
function coral(S::Matrix, T::Matrix)
    covS = cov(S, dims=2) + I(size(S, 1))
    covT = cov(T, dims=2) + I(size(T, 1))
    Shat = covS^(-0.5) * covT^(0.5)
    Shat = (Shat' * S)
    return Shat
end
```

Figure 4.3 CORAL Domain Adaptation - S stands for Source, and T is for Target.

The objective function of CORAL aims to minimize the difference in the covariance matrices of the source and target domain features. Let S and T represent the feature matrices for the source and target domains, respectively. The covariance matrices for S and T are denoted as cov_S and cov_T , respectively. The CORAL objective function can be expressed as in the code piece at 4.3. The goal is to align the source domain data S to the target domain data T by minimizing this difference.

In practice, this can be achieved by transforming the source features S to $\hat{S}$ such that the covariance of $\hat{S}$ is close to C_T . This transformation can be done through various methods, including linear transformations or more complex approaches using deep learning (Sun, Feng, Saenko 2017). For our experiments, we used the linear transformation approach as described in code 4.3.

4.2.2 Maximum Mean Discrepancy (MMD)

Maximum Mean Discrepancy (MMD), as delineated by (Gretton et al. 2006), is a statistical measure extensively employed in machine learning, predominantly in the realms of domain adaptation and kernel methods. The fundamental objective of MMD is to quantify the disparity between two probability distributions, usually pertaining to the source and target domains in domain adaptation scenarios. This quantification is achieved by contrasting the means of these distributions within a high-dimensional feature space. MMD's utility is particularly pronounced in instances where there is a need to ascertain whether two samples originate from distinct distributions, all the while refraining from making presuppositions about the nature of these distributions.

```
euclidean(X, Y) = pairwise(Euclidean(), X, Y) # returns n x n matrix

radialBasisFunction(X, Y; bandwidth=.25) = exp(-euclidean(X, Y)^2 / (2 * bandwidth^2))

function mmd(S::Matrix, T::Matrix)
    XX = radialBasisFunction(S, S)
    YY = radialBasisFunction(T, T)
    XY = radialBasisFunction(S, T)
    return mean(XX - 2XY + YY)
end
```

Figure 4.4 MMD with Gaussian Kernel for Domain Adaptation

As a kernel-based methodology, MMD allows for the utilization of a diverse array of kernel functions, tailored for domain adaptation tasks. The efficacy of each kernel type is contingent upon the specific attributes of the data and the requirements of

the task at hand. Factors such as the need to account for non-linear relationships, the interplay among features, and resilience to various kinds of noise or anomalies play a pivotal role in kernel selection. In our empirical studies, we have opted for the Gaussian kernel, also known as the radial basis function (RBF) kernel, owing to its recognition as one of the most versatile and broadly applicable kernel functions in this context (code at 4.4) .

4.3 Using Pivot Language for Cross Model Alignment

In the experiments conducted for Cross-model alignment, we methodically try to align GloVe_{en} (source) with FastText_{it, de, es, fi} (target), using alignment techniques in the VecMap framework, without employing domain adaptation techniques. This framework encompasses not only unsupervised alignment techniques but also supervised and semi-supervised ones.

In the unsupervised alignment technique, the words with the highest frequency between source and target languages are utilized for seed dictionary initialization. Subsequently, the alignment is executed through the updating indices of words used in calculating the rotation matrix in each iteration.

In the semi-supervised alignment technique, common numerals in the languages involved in the alignment process are utilized for seed dictionary initialization, followed by the process using the unsupervised alignment technique. Finally, in the supervised alignment technique, as the name suggests, the alignment process is conducted using a pre-prepared dataset specific to the source and target language pair.

Our results obtained using the unsupervised alignment method revealed that the alignment process was successful only between a few language pairs in cross model embeddings. This is contrary to "homogeneous" model alignment where same embedding models align with each other. For instance different language pairs of word embeddings provided by VecMap (or MUSE) framework align with each other (figure 4.5). This led us to initially try to find a proper seed dictionary initialization with a proper number of word involvement. Figure 4.6 shows the results for different seed initialization with different size of words for alignment operation for cross model alignment. In the figures provided, we present accuracy results considering various parameters. These parameters include different seed sizes and the number of words utilized during the alignment process. The figures titled *Accuracy & Seed Size* depict the average accuracy results with different word counts used for rotation. Conversely, the figures labeled *Accuracy & Embedding Size* demonstrate the impact of varying seed sizes on the average accuracy of the alignment.

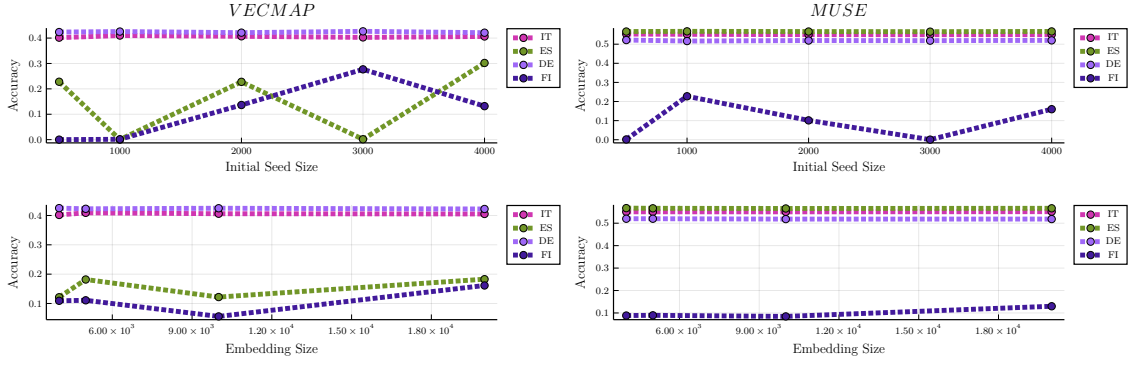


Figure 4.5 Comparisons of word alignments within VecMap and MUSE embedding models for different language pairs.

This misalignment situation between different word embedding models led us to try the semi-supervised alignment technique. However, the results from these experiments were inferior to those obtained from the unsupervised method, and no success was achieved between any language pairs. Consequently, to ascertain the effectiveness of the alignment process, we employed the validation sets of each language pair using 200, 400, and 600 word pairs to check whether the supervised alignment process would succeed between different embedding models. The outcomes of this experiment indicated the feasibility of conducting alignment processes between different models. The results of the supervised alignment processes between Glove_{en} and FastText models are shared in tables 4.1 and 4.2.

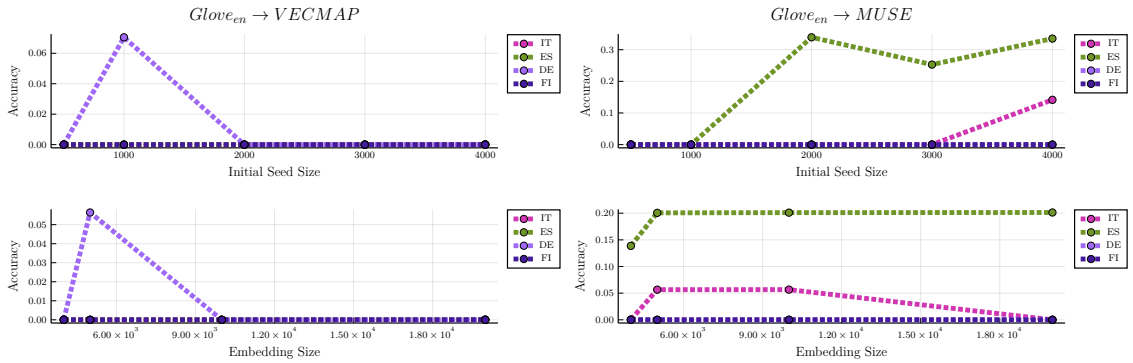


Figure 4.6 Comparisons of word alignments of Glove_{en} embeddings within VecMap and MUSE embedding models for different language pairs.

The outcomes from our supervised experiments indicate that the alignment process can be successfully conducted between different models, regardless of the language pair. This finding shifts our focus to exploring methods for achieving consistent results in unsupervised alignment. Utilizing the training set for alignment process implementation is the most robust method. However, creating a training set for each

word pair is not a feasible task. Therefore, conducting the alignment process in an unsupervised manner, although more challenging, is a more realistic method.

Table 4.1 Supervised Glove to VECMAP embedding mapping results

Language (Coverage)	GloVe ₂₀₀	GloVe ₄₀₀	GloVe ₆₀₀
IT (1490/1500)	.016	.06	.09
DE (1486/1500)	.024	.086	.14
ES (1491/1500)	.031	.069	.1
FI (1414/1500)	.02	.063	.08

Table 4.2 Supervised Glove to MUSE embedding mapping results

Language (Coverage)	GloVe ₂₀₀	GloVe ₄₀₀	GloVe ₆₀₀
IT (1487/1500)	.02	.07	.10
DE (1484/1500)	.01	.05	.08
ES (1490/1500)	.03	.09	.12
FI (1388/1500)	.005	.01	.02

The results obtained from unsupervised alignment experiments between compatible models have been observed to be successful. Leveraging this, we can use word embedding models with a common language between models known to be incompatible for domain adaptation, thus making the incompatible model compatible. For example, GloVe_{en} and MUSE_{de} cannot align with each other. In this case, our first step should be to use MUSE_{en} to adapt the domain of GloVe_{en}. This approach allows us to execute the cross-model alignment process in an unsupervised manner.

4.3.1 CORAL

In the application of the CORAL (Correlation Alignment) method for domain adaptation, we utilized the indices of a shared vocabulary that is common to both the source and target embedding spaces. This common vocabulary serves as the foundational element in aligning the two domains. By employing an orthogonal mapping technique, we then strategically manipulated the source domain embeddings to more closely resemble those of the target domain. This 'rotation' of the source domain towards the target domain is a critical step in the CORAL approach.

The primary objective of this method is to reduce the discrepancies between the source and target distributions. By focusing on the second-order statistics, which involve the covariances of the distributions, the CORAL method effectively minimizes the domain shift. This alignment is crucial, as it ensures that the patterns and structures inherent in the source domain are adjusted to mirror those in the target domain more closely, thereby facilitating a smoother and more accurate domain adaptation process.

4.3.2 MMD

In the application of the Maximum Mean Discrepancy (MMD) method, our experiment employed a neural network (NN) configuration. Recognizing the need for a balance between capturing the intricate distributional characteristics of different domains and maintaining computational efficiency, we opted for a straightforward, single-layer neural network design. This streamlined approach was pivotal in ensuring that the network could effectively analyze and adapt to the distributional variations between domains without excessive computational demands.

Similar to the strategy used in the CORAL method, this approach also leverages a common vocabulary extracted from the source and target embedding spaces. In the context of MMD, this shared vocabulary plays a crucial role. It serves as a basis for training the neural network, providing a consistent linguistic framework across both domains. This training enables the neural network to learn and minimize the MMD loss, which is essential for measuring and reducing the disparities between the source and target domain distributions.

Once the network is trained, the source domain embeddings are processed through it. This step is critical as the trained network applies its learned alignment, guided by the minimized MMD loss, to adjust the source embeddings closer to the target domain's distributional structure. Such an alignment is key to achieving effective domain adaptation, as it ensures that the source domain's data representation becomes more compatible with that of the target domain, thus facilitating better transfer learning or cross-domain applications.

CROSS MODEL KNOWLEDGE DISTILLATION IN SENTENCE EMBEDDING MODELS

In this chapter, we delve into the transfer of knowledge between two distinct sentence embedding models. It should be noted that a significant portion of the research, experiments, and findings presented in this chapter were originally published in Applied Sciences, MDPI, 2023, detailed in the 'Published Papers' section of this thesis.

The first model is characterized by its high performance and computational intensity, while the second is a more streamlined model, primarily utilizing word vector averaging. Our objective is to augment the capabilities of the streamlined model by harnessing the sophisticated features of the high-performance model. Diverging from conventional knowledge distillation techniques that typically focus on aligning logits or hidden states between teacher and student models, our methodology concentrates on aligning the word vector representations of the student model with the teacher model's output sentence vectors. To achieve this, we implement two distinct strategies: minimizing distance and a hybrid approach combining distance reduction with perplexity minimization. We use WMT datasets for training purposes and assess the enhanced embeddings' efficacy through tasks from Google's Analogy and Meta's SentEval datasets. The outcomes demonstrate that our proposed models effectively preserve and convey information, each in a manner particular to its design.

5.1 Cross Model Alignment with Knowledge Distillation

In this part of the thesis, we delve into the intricacies of knowledge distillation, with a specific focus on the transfer of insights between two distinct sentence embedding models: FastText and sBert. Knowledge distillation typically entails a complex 'teacher' model imparting its knowledge to a simpler 'student' model. However, challenges arise due to the resource demands of this process, and the possibility that the student model might not fully capture the predictive strengths of the teacher model.

We particularly address challenges that emerge when limited access to model parameters exists, or when there's a necessity to redevelop models within the same programming language and framework. The disparate nature of FastText and sBert – in terms of their architectural designs, training datasets, and the dimensions of their output vectors – presents unique hurdles. FastText, employing a bag-of-words method, offers the advantage of rapid sentence embedding generation without requiring GPU support, though it often sacrifices accuracy. In contrast, sBert leverages a transformer architecture to produce sentence embeddings, yielding enhanced performance in various NLP tasks but at the expense of speed due to greater computational demands.

The primary aim of this chapter is to construct models that effectively harmonize word-level and sentence-level representations, effectively linking the word embeddings of FastText with the sentence vectors of sBert. The models we propose are designed to produce closely aligned sentence representations. To facilitate the transfer of information, we utilize the WMT EN-ES dataset and employ classification tasks and semantic similarity evaluations from the SentEval datasets as validation tools.

5.2 Methodologies

FastText and sBert, each with their unique methodologies, generate distinct sentence embeddings, resulting in different data representations for identical sentences. sBert incorporates contextual nuances by considering word positions during embedding generation, whereas FastText, using a bag-of-words approach, does not account for word order, leading to a loss of positional information. This fundamental distinction results in varied representations of the same sentence in their respective spaces. Despite these differences, both models strive to extract meaning from sentences, suggesting potential correlations between their vector representations in respective spaces (Figure 5.1). To enhance this correlation, we propose two methodologies:

1. **Distance Minimization** focuses on reducing the dissimilarity between sBert and FastText embeddings by modifying their representational orientations.
2. **Distance & Perplexity Minimization** not only minimizes the gap between sBert and FastText but also verifies the accuracy of contextual information along with the distance.

These approaches aim to enhance FastText's word-level representations by aligning them with sBert's embeddings. Both strategies apply representation distillation methods, akin to those in (Sanh et al. 2020), utilizing only sentence embeddings from

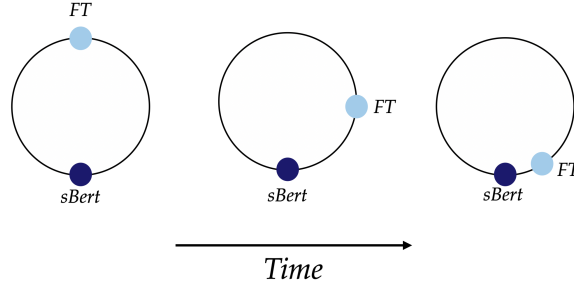


Figure 5.1 Illustration of the distance minimization process. The two vectors represent sentence embeddings from FastText (light blue) and sBert (dark blue) for the same sentence. The goal is to minimize their angular distance, aligning FastText’s sentence representations closer to those of sBert.

sBert. The overarching objective is to ensure that the refined embeddings align closely with the benchmark quality of the sBert model. After this distillation process, the revised FastText word embeddings will be utilized to create sentence embeddings via a bag-of-words methodology for sentence evaluation tasks, as per the original FastText framework.

Our models are developed in accordance with their respective objective functions. The distance minimization approach employs an encoder architecture, while the second model, which also evaluates the accuracy of contextual information in sentences, utilizes a decoder architecture due to its reliance on cross-entropy loss for minimizing perplexity. To achieve more nuanced decoding, an encoder architecture precedes the decoder, forming a sequence-to-sequence model.

5.2.1 Distance Minimization

This method aims to bridge the gap between sentence embeddings across diverse vector spaces. Utilizing a transformer encoder architecture, we focus on minimizing the generated sentence vectors through regression. Our exploration includes the integration of two distinct distance metrics as loss functions: Euclidean distance and cosine similarity. By combining these metrics in the loss function, we capture vital information that might be overlooked by using either metric alone.

In this process, FastText word embeddings are refined by leveraging the output vectors from the sBert model for equivalent sentences. This approach enhances the informational richness of FastText embeddings, thereby achieving a closer alignment with the sBert embeddings within the targeted space. Figure 5.2 illustrates the proposed model architecture, which encompasses a word embedding layer alongside a transformer encoder structure. The encoder is tasked with generating embeddings

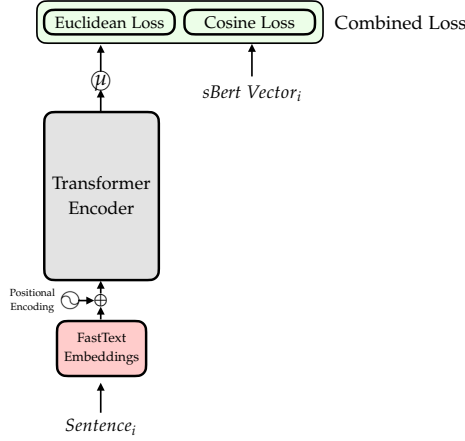


Figure 5.2 The Distance Minimization model includes a word embedding layer for refining FastText word embeddings and a transformer encoder to project the outputs. The model utilizes a combined loss function to measure the distance relative to sBert sentence vectors.

for each word within a sentence. In each processing batch, the encoder outputs a tensor with the dimensions $D \times L \times B$, where D is the dimensionality, L the sentence length, and B the batch size. We then compute the average of these tensors, resulting in an output tensor of dimensions $D \times B$. The loss is subsequently calculated using both Euclidean and cosine distances, each weighted equally.

5.2.2 Distance & Perplexity Minimization

In the context of knowledge distillation, we often conceptualize the process as transferring insights from a 'teacher' model to a 'student' model, with a focus on aligning their entropy levels. In natural language processing, this alignment is frequently interpreted through the lens of perplexity, a metric used to evaluate how effectively a probabilistic model predicts a sample. Notably, sBert exhibits more advanced representational capabilities than FastText, thereby offering superior predictive accuracy.

Our approach to enhancing information transfer involves employing a sequence-to-sequence transformer model, as illustrated in Figure 5.3. In this setup, the word embedding layers are common to both the encoder and decoder components. The encoder initially processes input sentences, creating a series of word representations based on their contextual positions. These are then fed into the decoder's cross-attention module. Here, for every target sequence position, the decoder assesses the relationship between the encoder's output (key-value pairs) and its queries to predict the next word in the sequence, utilizing cross-entropy loss

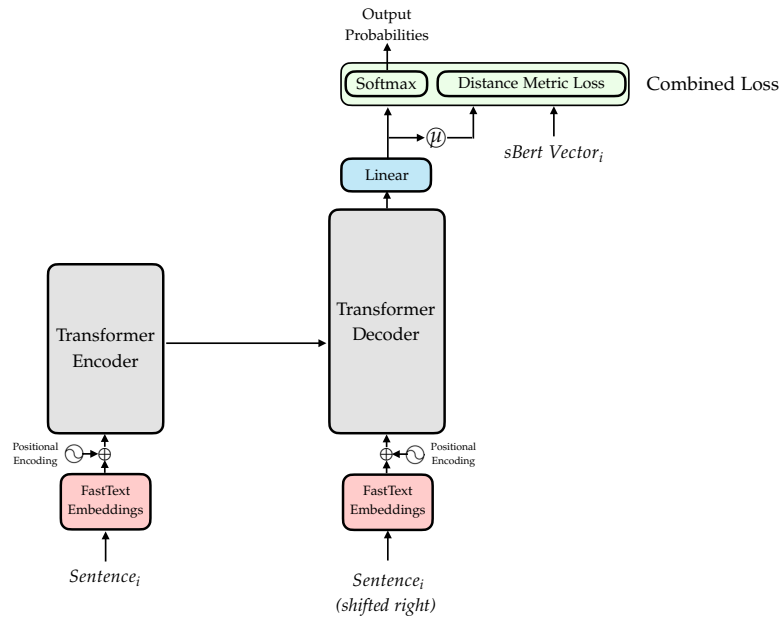


Figure 5.3 The Distance Perplexity Minimization model, featuring a sequence-to-sequence design with a shared embedding layer. The model utilizes FastText word embeddings and aims to preserve contextual word information, enhancing sentence perplexity for improved representations. This leads to more thoughtful reflection of knowledge in the word embedding layers.

during training to minimize perplexity. The decoder ultimately produces a set of word representations for the output sequence, which are averaged to form sentence vectors. These vectors are then compared against their corresponding sBert-generated counterparts, ensuring a comprehensive alignment process.

6

EXPERIMENTS

In this chapter, we present a comprehensive overview of our experimental setup. Detailed information is provided on our approaches to cross-model alignment between different word embedding models in Section 6.1. Additionally, we explore cross-model alignment using distillation for different sentence embedding models in Section 6.2. Both sections include in-depth descriptions of the data used and the specifics of the models implemented, offering a thorough understanding of our methodologies and the frameworks underpinning our research.

6.1 Experiments on Cross Model Alignment in Word Embedding Models

6.1.1 Data

In this subsection, we delve into the specifics of our research’s data components. We will detail the word embedding models employed in our analysis and outline the validation datasets used for evaluating the performance of the models we obtained. This subsection aims to provide a comprehensive overview of the data framework that underpins our research, offering insights into both the linguistic tools and the evaluative metrics that form the foundation of our study.

6.1.1.1 Word Embeddings Data

Different word embedding models employed in our cross-model alignment experiments are detailed in Table 6.1. All embeddings utilized in this study are sourced from publicly available datasets provided by various organizations and share a common feature in their representation space: each model uses a 300-dimensional space to represent a single word. This uniformity in dimensionality is crucial for ensuring consistency and comparability across different embedding models.

Specifically, the GloVe embeddings, comprising 400 thousand unique words, are

derived from the Wikipedia 2014 dataset. These embeddings are known for their comprehensive coverage and are particularly useful for tasks involving English language data.

Additionally, the MUSE word embeddings, which are made publicly available by Meta (formerly known as Facebook), offer an extensive range of word embeddings for 294 languages. For the purposes of our experiments, we focused on English, Italian, German, and Finnish embeddings from MUSE. This selection was guided by the need to align with the languages available in the VecMap embeddings, ensuring consistency and comparability across our experimental setups.

Furthermore, the word embeddings provided by the VecMap framework are also publicly accessible. VecMap is recognized for its ability to align pre-existing word embeddings into a shared space, facilitating cross-lingual applications and comparisons. This feature of VecMap is particularly valuable in our study, enabling the effective alignment and analysis of embeddings across the chosen languages.

Table 6.1 Different word embedding models that are used for cross model alignment

Model	Language	Voc. Size	Dims
GloVe	English	400 Thousand	300
MUSE	English	2.5 Million	300
	Italian	871 Thousand	
	German	2.2 Million	
	Spanish	985 Thousand	
	Finnish	730 Thousand	
VecMap	English	200 Thousand	300
	Italian	200 Thousand	
	German	200 Thousand	
	Spanish	200 Thousand	
	Finnish	200 Thousand	

6.1.1.2 Validation Data for Alignment

To validate our models, we utilized the public validation set provided by the VecMap framework. This validation set is a crucial component of our methodology as it offers a standardized basis for evaluating the performance and accuracy of our models in the literature.

The datasets comprise 1,500 source-target word pairs, meticulously curated to represent a wide range of linguistic relationships. An important aspect of this dataset is that each source word is associated with multiple target translation words. This feature of the dataset allows for a more comprehensive assessment of the models' capabilities in capturing the nuances of language translation and semantic relationships. By accommodating multiple possible translations for each source

word, the dataset provides a more realistic and challenging validation environment, reflecting the complexity and variability inherent in natural language. Each obtained model is evaluated based on the accuracy of matches between source and target words in these validation datasets.

The use of these datasets in our validation process ensures that our model assessments are grounded in diverse and representative linguistic scenarios, enabling a thorough and robust evaluation of the models' alignment and translation effectiveness across different languages.

6.1.2 Experiment Procedure

Experiments for cross domain alignment focused on domain adaptation between source and target domains using shared vocabularies from common (pivot) languages as a supervised signal. The main objective is to minimize the domain differences between the source domain to the target domain effectively. After obtaining the domain adapted source embedding space, the unsupervised word alignment procedure is carried out with a specific focus on varying the dimensions involved in the seed dictionary size and the number of words that are used for finding the rotation matrix (for more information see subsection 6.1.2.3).

The domain adaptation experiments utilized a common vocabulary set found in the pivot language. For all the experiments, the pivot language is the English word embedding space. A total of 80,000 words from the shared vocabulary were employed as the foundational dataset for the study. This selection was intended to provide a robust and varied linguistic foundation for the domain adaptation process. This shared vocabulary is divided into training and test sets according to .85/.15 ratio. In all of our experiments, both source and target spaces are unit, and centered normalized in order to facilitate the domain adaptation and alignment process. All experiments are executed on NVIDIA RTX-A4500.

6.1.2.1 Coral Experiments

For the CORAL method, the indices of the common vocabulary for source and target embedding spaces are used for domain adaptation. Then the same vocabulary is applied for rotating the source domain towards target domain by orthogonal mapping technique. This approach aimed to align the source and target distributions by minimizing domain shift, primarily focusing on the second-order statistics.

6.1.2.2 MMD Experiments

For the Maximum Mean Discrepancy (MMD) method, a neural network (NN) setup was utilized. The experiment opted for a simple, single-layer neural network. This choice was governed by the need to effectively capture the distributional characteristics of the domains while maintaining computational efficiency. As in the CORAL method, the same extracted common vocabulary from source and target embedding spaces are used for training. Then the source embedding space is passed to the obtained trained model.

In the training phase of our neural network, we employed a dense layer configuration, specifically a 300 by 300 layer, integrated with a tanhshrink activation function. This function was chosen for its ability to introduce non-linearity while effectively managing the vanishing gradient problem, making it suitable for complex domain adaptation tasks.

The optimization of the network was handled using the AdamW optimizer. AdamW, a variant of the Adam optimizer (Loshchilov, Hutter 2019), incorporates weight decay, which helps in better regularization and potentially leads to more effective training outcomes. We set the batch size to 1000, a value determined to balance computational efficiency and the need for gradient accuracy.

A key component of our training process was the implementation of a learning rate scheduler. We initiated the training with a learning rate of $1e-1$, and over time, it was decayed to reach $1.1e-2$. This approach allowed for a more dynamic adjustment of the learning rate, enabling the network to initially make larger updates to the weights and gradually refine these adjustments as the training progressed.

Our training regimen was specifically tailored for two distinct domain adaptation tasks: adapting Glove embeddings to Vecmap and Glove to Muse. For the adaptation to Vecmap, the network underwent 42 epochs of training. This duration was determined to be optimal for achieving sufficient convergence without overfitting. In the case of Glove to Muse adaptation, the training extended for 50 epochs, a decision guided by the different characteristics of the Muse dataset and the need for a more extended period to achieve optimal adaptation.

6.1.2.3 Cross Model Alignment Experiments

After adapting the $Glove_{en}$ source embedding space to the target domain ($FastText_{en}$ version of MUSE or VecMap), we conducted unsupervised alignment experiments with a specific emphasis on varying the dimensions involved in both the seed and rotation

processes within the embedding spaces. This approach was crucial for gaining insights into how different dimensional configurations influence alignment outcomes.

In these experiments, we varied the size of the dictionary seed, which serves as the initial starting point for the alignment process, across different experimental runs. This variation was integral in evaluating the robustness and consistency of the alignment technique under varying initial conditions. By experimenting with different seed sizes, we were able to discern the impact of various starting points on the ultimate alignment results. Specifically, the initial seed sizes used in our experiments ranged from the first 500 to 4,000 words (doubled seed size) from both the source and target embedding spaces.

In a similar vein, the rotation process within the embedding space was also subject to dimensional variation. Being a pivotal step in aligning embedding spaces across different languages in an unsupervised manner, the number of elements included in rotation plays a key role. We explored how altering the size of the word set used for rotation – experimenting with the first 4,000, 5,000, 10,000, and 20,000 words – could affect the alignment’s effectiveness. This approach allow us to determine the most suitable rotation matrix for the alignment process.

These dimensions, both in terms of seed size and rotation, were strategically chosen to provide a comprehensive understanding of the behavior and performance of the unsupervised alignment method under diverse conditions. The results from these experiments are expected to offer valuable insights into the optimization of unsupervised alignment techniques in natural language processing and related fields.

6.2 Experiments on Cross Model Knowledge Distillation in Sentence Embedding Models

6.2.1 Data

In our experimental framework, we utilize three distinct data sets. The initial data set is designated exclusively for the process of transferring information. The second set is employed to analyze the alterations in the updated word embeddings in relation to the original versions and to evaluate the performance of the proposed models. The final data set is used to gauge the effectiveness of the transferred sentence embeddings, specifically in comparison with the sBert sentence embeddings. For consistency and focus in our tasks, we exclusively utilize English datasets across all experimental activities.

6.2.1.1 Data for Transferring Knowledge

In order to enable the transfer of knowledge from sBert to FastText sentence embeddings, we used the widely available WMT dataset, which includes around 2 million sentences. For consistency in comparison, the same randomly chosen sentences were used in both the training and testing phases across all the models we proposed. The training subset comprised 285 thousand sentences selected at random, while the test subset included 15 thousand sentences, making a total of 300 thousand sentences. This resulted in a vocabulary of 56 thousand words derived from the training sentences, ensuring that all evaluations were conducted on a specific subset of the FastText word embeddings.

6.2.1.2 Word Level Evaluation: Word Analogy Tasks

The Google Analogy Test Set, as compiled by (Mikolov, K. Chen, et al. 2013), consists of word pairings designed to evaluate the effectiveness of word embeddings in capturing various relationship types. Given that our models are intricately involved with word embeddings, demonstrating the impact of knowledge transfer on the enhanced embeddings relative to their original versions is crucial.

Our evaluation of this test set primarily revolves around two key tasks: determining both syntactic and semantic connections between word pairs. The objective of these tasks is to accurately identify the corresponding word for a specific query, guided by a set of provided key-value pairs.

6.2.1.3 Sentence Level Evaluation

SentEval, a toolkit developed by Meta (previously known as Facebook) (Conneau, Kiela 2018), serves as an essential resource for evaluating sentence embedding models. This toolkit, known for its wide recognition as a benchmarking standard in sentence representation tasks, offers a variety of tasks and linguistic probes.

Within SentEval, there are several distinct tasks such as classification, semantic relatedness, probing, and image caption retrieval. The classification task is composed of eight different datasets, covering analyses like sentiment analysis, question type classification, and discerning between subjective and objective content. The semantic relatedness task evaluates how closely or distantly a pair of sentences is related. Given our research focus on the interplay between different embedding models, we will primarily analyze the results from the classification and semantic textual similarity tasks, opting to exclude probing and image caption retrieval tasks from our analysis.

6.2.1.4 Classification Tasks

The datasets listed below are intended for the development of a logistic regression classifier utilizing sentence embeddings.

- *CR* (Customer Review): This dataset is composed of product reviews from customers. Each piece of feedback is categorized according to the sentiment conveyed, with labels indicating positive or negative. The objective is to predict the sentiment reflected in customer product reviews.
- *MR* (Movie Review): This dataset features movie reviews. Each review is labeled as having either a positive or negative sentiment. The task involves predicting the sentiment of movie review.
- *MPQA* (Opinion Polarity): This dataset features annotations of opinion polarity within phrases from newswire texts. The challenge lies in classifying each phrase according to whether it conveys a positive, negative.
- *SUBJ* (Subjectivity Status): This collection consists of sentences derived from movie reviews and plot summaries. The goal is to determine whether each sentence expresses personal opinions and feelings (subjective) or states factual information (objective)
- *SST-2* (Stanford Sentiment Treebank): This sentiment analysis resource provides sentences extracted from movie reviews, with the objective being to determine if the sentiment is positive or negative.
- *SST-5* (Stanford Sentiment Treebank): An extension of SST-2, this dataset encompasses a spectrum of sentiment annotations with five distinct levels, from very negative to very positive. It facilitates a more nuanced sentiment analysis task by allowing for fine-grained classification of emotions.
- *TREC* (Text REtrieval Conference): A dataset curated for the task of question classification, it differentiates between six specific types of questions, including categories like location, numeric value, and person.
- *MRPC* (Microsoft Research Paraphrase Corpus): is a collection of sentence pairs automatically extracted from online news sources, with annotations to indicate whether each pair captures the same semantic content, thereby serving as a resource for identifying paraphrases in parallel news articles.

6.2.1.5 Semantic Relatedness Tasks

SentEval, apart from its other functionalities, includes tasks focused on semantic relatedness, categorized into two groups based on their evaluation methods: supervised and unsupervised models. The aim of both categories is to assign a semantic similarity score ranging from 0 to 5 to sentence pairs, mirroring scores previously assigned by human evaluators. A significant aspect of these tasks is to determine how well the cosine similarity of sentence embeddings aligns with human judgments. The efficacy of both supervised and unsupervised models in these tasks is determined through the use of Pearson and Spearman correlation coefficients.

The datasets ranging from STS12 to STS16 encompass sentence pairs from diverse sources like news articles, online forums, and descriptions from multimedia content. Furthermore, the STS Benchmark (STSB) assembles English datasets used in the Semantic Textual Similarity (STS) challenges at the SemEval workshops from 2012 to 2017. This collection includes texts from various domains, such as image captions, news headlines, and user-generated content on forums. Additionally, the SICKR dataset (Sentences Involving Compositional Knowledge for Relatedness) serves as a standard for assessing model capabilities in interpreting sentence meanings in terms of semantic relatedness. These datasets collectively provide a robust platform for evaluating computational models on text similarity tasks.

6.2.2 Experiment Procedure

This section outlines the experimental setup designed to assess the effectiveness of our proposed methods. Given the task of transferring information between two distinct models, we aimed to minimize the learnable parameters. Rather than training the network to adapt to the dimensional differences between sBert sentence vectors and FastText sentence (word) vectors, we employed an orthogonal mapping technique. This approach reduced the dimensions of sBert sentence vectors to match those of FastText vectors¹, a strategy akin to supervised dimensionality reduction (Hardoon, Szedmak, Shawe-Taylor 2004). The vectors obtained through this process are referred to as sBert_{Reduced} vectors, with their test values presented in the results section.

The distance minimization method focuses on knowledge transfer by aligning the sentence embeddings from the encoder with sBert sentence vectors, using a combined loss function of Euclidean and cosine distances, weighted equally. Conversely, the distance & perplexity minimization experiment highlights the benefits of including

¹sBert sentence vectors have 768 dimensions whereas FastText embeddings have 300 dimensions. This situation forces the proposed models to output 300 dimensional vectors which requires to have an additional feed forward network for eliminating the dimensional difference.

Table 6.2 The validation results for *distance minimization* and *distance & perplexity minimization* methods.. The *distance minimization* approach uses combined loss of euclidean and cosine distances. Whereas the *distance & perplexity minimization* method uses combined loss of cosine and cross entropy. Each method uses equal weights between their loss functions.

Model	Objective	Combined Loss	Cosine	Euclidean	XE
Distance Min	EUC + COS	0.38	0.21	0.58	-
Distance&PP Min	XE + COS	0.08	0.16	-	0.013

contextual information to improve the transfer between sBert sentence vectors and FastText word embeddings. This method differs by incorporating a decoder substructure.

In both distance minimization and distance & perplexity minimization methods, we developed single-layer architectures with the following specifications: The embedding layers used 300-dimensional vectors from FastText. Each model included 4 attention heads and produced 300-dimensional sentence vectors for comparison, paralleling the 300-dimensional projected space of sBert representations. Optimal training parameters were identified through extensive hyperparameter optimization, revealing a consistent learning rate of $3e-4$ across all models. This rate, combined with the Adam optimizer and a batch size of 128, yielded the most effective training results.

For a fair evaluation, we randomly selected 300 thousand sentences from the WMT corpora, dividing them into a training set of 285 thousand sentences and a test set of 15 thousand. The validation outcomes, detailed in Table 6.2, indicate that the distance & perplexity minimization method achieves superior cosine similarity results.

Both methods can generate sentence vectors in two distinct manners: by averaging the updated FastText word embeddings (bag-of-words approach) or directly through the model. While the updated FastText embeddings will be assessed using word analogy and SentEval datasets, model outputs will be exclusively evaluated on sentence evaluation tasks.

7.1 Results for Cross Model Alignment in Word Embedding Models

7.1.1 CORAL

In the figures presented in 7.1, we illustrate the accuracy outcomes after we apply CORAL domain adaptation between $Glove_{en}$ and FastText ($VecMap_{en}$ and $MUSE_{en}$) under common vocabulary. Our experiments are influenced by a range of parameters, notably various seed sizes and the quantity of words utilized in the alignment process. These figures summarize the averaged results from 20 distinct experiments conducted for each language alignment. Specifically, the charts labeled *Accuracy & Seed Size* reveal the correlation between the size of word groups used for rotation and the resultant average accuracy. In contrast, the *Accuracy & Embedding Size* charts demonstrate the effect of different seed sizes on the average accuracy of the alignment operation.

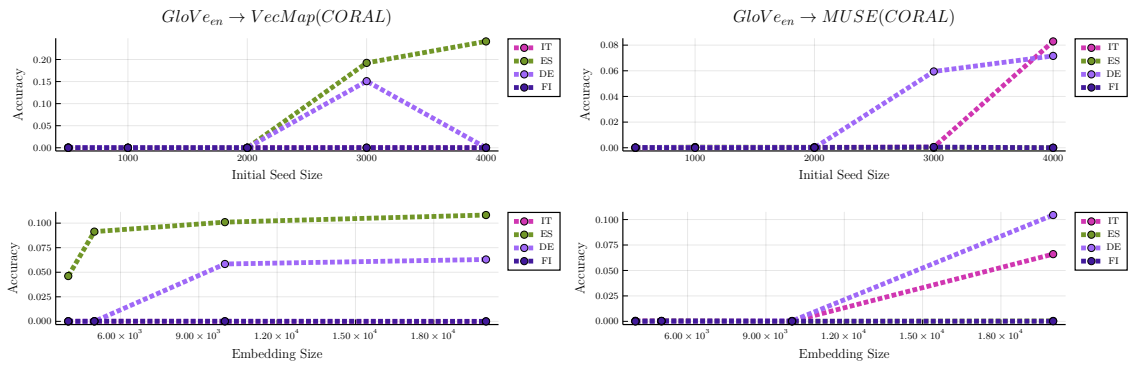


Figure 7.1 Comparison of CORAL domain adaptation effects on $Glove_{en}$ and FastText (VecMap & MUSE) embeddings.

These results distinctly reveal that the alignment operation between different word embedding models lacks the consistency observed within their internal alignments, as demonstrated in figure 4.5. Applying CORAL domain adaptation not only alters the success rates of alignments but also affects the aligned language pairs. Significantly,

the application of CORAL tends to bias the accuracy results in favor of VecMap embeddings, while concurrently reducing the alignment success of MUSE embeddings. This phenomenon is clearly evidenced in figure 4.6.

7.1.2 MMD

Despite the suboptimal results obtained in our MMD (Maximum Mean Discrepancy) experiments, it is crucial to demonstrate the thoroughness of our training process. This section aims to present a transparent view of our MMD model’s performance during training and validation phases, highlighting the efforts made towards optimizing the neural network.

The initial maximum mean discrepancy scores for training and validation sets between source and target domains is given in table 7.1. The figures in 7.2, we provide a detailed graph depicting the training and validation losses over the course of the experiment. This graph is essential for understanding the model’s behavior during the training process and provides insights into the challenges encountered in minimizing the loss.

Table 7.1 MMD loss values between source and target domains. These values are obtained without model usage.

Source $\rightarrow$ Target	Training	Validation
Glove _{en} $\rightarrow$ VecMap _{en}	0.083	0.085
Glove _{en} $\rightarrow$ MUSE _{en}	0.126	0.128

The graphs in Figure 7.2 depict the progression of training and validation losses over various epochs. Notably, the training losses for both experiments initially exhibit slight fluctuations, eventually stabilizing into consistent decreases. This indicates effective learning from the training data. In contrast, the validation losses demonstrate significant fluctuations, suggesting a divergence between training and validation performance. Such pronounced variability in validation losses points to the inherent complexity of the task at hand. We still managed to decrease the validation losses close to the training losses.

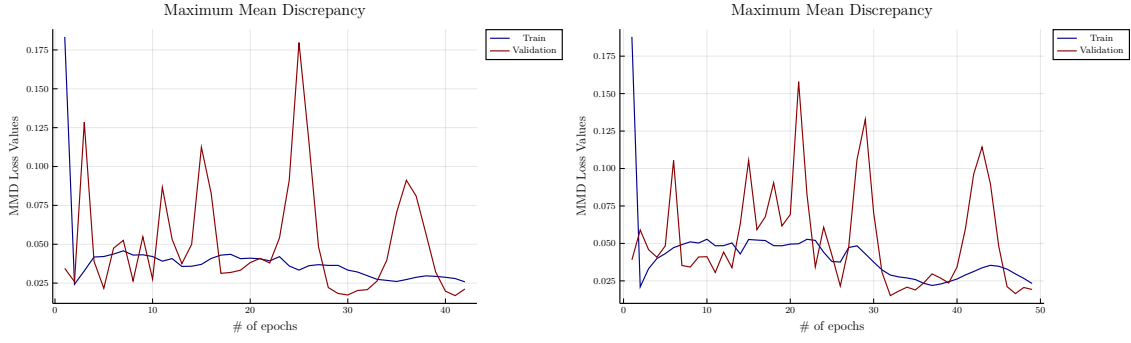


Figure 7.2 Left represents the MMD loss values between $Glove_{en}$ and $VecMap_{en}$, whereas the right represents the $Glove_{en}$ and $MUSE_{en}$. Blue lines indicate the training losses, red lines indicate the validation loss values.

These pronounced fluctuations in validation losses not only reflect the difficulty in achieving model generalization but also have a direct impact on the poor results observed in the alignment process between cross models. Despite attempts to mitigate this issue through various adjustments — including increasing the size of the validation set, enhancing batch sizes, and experimenting with different learning rates — the pattern of fluctuating validation losses persisted. This persistence highlights the challenge in achieving model generalization under the given task conditions and suggests the necessity for further investigation.

7.2 Results for Cross Model Knowledge Distillation in Sentence Embedding Models

This section discusses the impact of newly proposed models on two key tasks: Word Similarity and SentEval. These tasks are designed to assess the effectiveness of updated FastText word embeddings and outputs from the new models. The analysis is focused on a selected subset of FastText embeddings, featuring 56 thousand unique words, to ensure a sufficient vocabulary range for NLP tasks and avoid skewing results due to the integration of new vectors.

For word similarity, two versions of updated FastText embeddings are introduced: one updated through the Distance Minimization method and the other through the Distance & Perplexity Minimization method. These represent the enhancements achieved by the respective models.

In sentence evaluation tasks, the models are categorized as follows: original sBert sentence embeddings are denoted as $sBert_{768}$; $sBert_{reduced}$ refers to sBert embeddings refined through orthogonal dimension reduction; and $sBert_{384}$ is a comparative

alternative sBert model. The Distance Minimization method is abbreviated as Distance Min, and the Distance & Perplexity Minimization method as Distance&PP Min. Both updated FastText versions are collectively labeled as FastText_{updated} in Tables 7.3 and 7.4. Additionally, the models' execution times are benchmarked against the baseline sBert₇₆₈ and original FastText embeddings.

7.2.1 Word Similarity Tasks

This section presents the results from evaluating the pre-trained FastText word vectors subset (FastText_{subset}) and their enhanced versions updated via the proposed models, as shown in Table 7.2. The table also includes results from sBert₇₆₈. For sBert, which employs sub-word tokenization, word embeddings are derived by averaging the sub-word vectors from its embedding layer.

Table 7.2 Results from the Google Analogy Task. sBert₇₆₈ shows averaged sub-word results, FastText_{subset} represents the original embeddings, and FastText_{updated} includes enhancements by the Distance Min and DistancePP Min methods.

Model	Semantic Acc.	Syntactic Acc.	Total Acc.
sBert ₇₆₈	30.77%	21.83%	22.01%
FastText _{subset}	98.72%	80.20%	80.58%
FastText _{updated} by Distance Min	1.28%	0.12%	0.14%
FastText _{updated} by Distance&PP Min	69.87%	52.67%	53.02%

The table reveals that the embeddings updated via the distance minimization method surprisingly lose all contextual information. Conversely, the distance & perplexity minimization method maintains this context. This discrepancy could stem from the differences in the underlying models: the former focuses on encoding sentences for distance comparison with sBert, while the latter uses encoded information for accurate predictions via cross-entropy. Additionally, the sBert vectors themselves could be a factor. The lower results from sBert-derived word vectors suggest that sentence vectors from sBert might distort the word embedding layers in the proposed models during training. As a result, the updated embeddings underperform compared to the original FastText_{subset} word embeddings.

7.2.2 Sentence Evaluation Tasks

This section details the results of the proposed models on the SentEval library. To ensure fair comparisons, the same parameters as those listed on the repository page were used. Classification tasks were conducted with a batch size of 64, using Adam as the optimizer, across 4 epochs and employing 10-fold cross-validation.

Different sBert sentence embedding models were evaluated to ascertain the

Table 7.3 Comparative performance of various models on classification tasks. The *Word Domain Coverage* row indicates how the classification accuracy correlates with the overlap between the training data from the WMT dataset and the vocabulary of various classification datasets.

Model	CR	MPQA	SST2	MRPC	MR	SUBJ	SST5	TREC
sBert ₇₆₈	86.86	89.32	88.74	74.07	85.05	93.97	49.19	94.0
sBert _{reduced}	85.56	89.18	88.85	75.42	84.89	93.45	48.46	92.0
sBert ₃₈₄	83.05	88.14	83.31	74.09	78.05	92.35	46.02	91.0
FastText _{subset}	76.88	87.49	80.07	71.71	75.25	98.93	42.76	78.4
FastText _{updated} w/ D. Min	66.31	83.17	70.95	67.94	67.15	81.48	33.98	66.4
Distance Min	73.22	85.3	68.75	68.23	67.61	82.85	35.93	60.4
FastText _{updated} w/ D&PP Min	74.97	88.06	80.51	70.67	73.76	89.17	41.0	77.2
Distance&PP Min	79.42	87.29	78.36	72.06	74.13	88.82	40.9	76.8
Word Domain Coverage	0.72	0.86	0.66	0.59	0.53	0.52	0.65	0.60

effectiveness of the dimensionality reduction technique, comparing the original sBert₇₆₈ model with a dimensionally reduced version (sBert_{reduced}) of 300 dimensions, and another model (sBert₃₈₄) with 384 dimensions on a 12-layer architecture. To evaluate sBert embeddings, a script from the Sentence-Transformers package was used, while FastText sentence embeddings were analyzed using a bag-of-words approach. For a broader comparison with other models, please refer to the benchmarks provided by (Reimers, Gurevych 2019), (Gao, Yao, D. Chen 2021).

Table 7.3 displays the performance of various models on classification tasks. The FastText word embeddings, refined through the distance minimization approach, show lower performance compared to the original FastText_{subset} across all evaluated tasks. Furthermore, using the distance minimization model in isolation doesn't enhance performance. In contrast, the second approach, distance & perplexity minimization, with its corresponding updates to FastText embeddings, yields better results on CR, MPQA, SST2, and MRPC datasets than FastText_{subset}. This improvement is partially explained by the Word Domain Coverage metric in the table, which evaluates the overlap between the training data's vocabulary and the vocabulary of the SentEval classification datasets. Generally, higher domain coverage scores, indicating greater vocabulary overlap, are linked to improved results. However, the MRPC dataset is an exception to this pattern. While most datasets focus primarily on building a classifier, MRPC requires comparing sentence pairs to establish semantic equivalence, which is a separate challenge. Thus, the enhanced performance on MRPC can be attributed not just to vocabulary overlap but also to the model's capacity to identify detailed semantic connections. This perspective is further corroborated by the insights from Table 7.4.

Table 7.4 provides the outcomes for the semantic textual similarity (STS) tasks. In line with the classification task results, the embeddings refined by the distance minimization method initially show weaker performance. However, a significant

Table 7.4 Results of Semantic Textual Similarity tasks on updated word embeddings and the original subset FastText word embeddings. Left values represent Pearson Correlation, whereas the right values present Spearman Correlation values.

Model	STS12	STS13	STS14	STS15	STS16	STSB	SICKR	Avg
sBert ₇₆₈	.72/.70	.78/.77	.80/.77	.82/.83	.82/.83	.82/.83	.87/.82	.80/.79
sBert _{reduced}	.72/.70	.78/.78	.80/.77	.82/.83	.82/.83	.83/.84	.87/.81	.81/.79
sBert ₃₈₄	.71/.70	.76/.77	.80/.77	.82/.83	.81/.82	.83/.83	.87/.80	.80/.79
FastText _{subset}	.32/.42	.39/.41	.46/.48	.53/.55	.39/.44	.61/.61	.76/.69	.49/.51
FastText _{updated} w/ D. Min	.29/.37	.30/.31	.38/.41	.42/.44	.31/.36	.47/.47	.64/.61	.40/.42
Distance Min	.47/.48	.55/.55	.54/.51	.59/.59	.54/.56	.50/.49	.59/.53	.54/.53
FastText _{updated} w/ D&PP Min	.33/.44	.39/.41	.46/.48	.52/.54	.39/.44	.57/.58	.74/.68	.49/.51
Distance&PP Min	.55/.57	.56/.58	.61/.58	.66/.66	.62/.63	.63/.62	.76/.67	.63/.62

Table 7.5 Execution times for sentence representation generation across different models.

Model	Time (secs)
sBert ₇₆₈	271.7
FastText _{subset}	23.46
Distance Min	88.4
Distance & PP Min	182.67

improvement is observed after further processing through the distance minimization model, with these enhanced embeddings surpassing the performance of the original FastText (FastText_{subset}) in a majority of the STS tasks. This indicates an overall higher mean performance. Notably, the embeddings updated by the distance & perplexity minimization method produce results similar to the original FastText_{subset}. Nevertheless, a considerable enhancement is seen in sentence pair evaluations using this model, outperforming FastText_{subset} in the majority of the tasks.

These results underscore that the effectiveness of the applied methods is more pronounced at the sentence level. This highlights the criticality of the selected model structure, loss function, and their congruence with the specific demands of the tasks.

7.2.3 Runtime Comparison

To demonstrate the effect of our proposed models on the generation of sentence embeddings, we compare their execution times. Each model was tested on a uniform dataset consisting of 300,000 sentences, which served as the basis for both training and validation. Apart from FastText, all models benefited from GPU acceleration and were run in 64-sentence batches. For generating sentence embeddings using the sBert₇₆₈ model, the Sentence-Transformers (Reimers 2019) Python package was employed.

Table 7.5 shows that the sBert₇₆₈ model has the longest processing time for generating sentence embeddings. On the other hand, FastText stands out for its efficiency, operating effectively even without GPU acceleration. The models we propose position

themselves as a middle-ground solution, balancing the computational resource requirements and time efficiency between these two extremes. Additionally, there is potential for further optimization in our models to enhance their execution times.



8.1 Cross Model Alignment on Word Embeddings

The exploration of bilingual and cross-lingual word embedding alignment has been a focal point for various research groups, each employing distinct methodologies. Predominantly, these methods have utilized orthogonal projection to facilitate unsupervised alignment across different languages, leveraging the most frequently occurring words in both the source and target languages for this purpose.

The study by (Hartmann, Kementchedjhieva, Søgaard 2018) marked a significant observation: unsupervised alignment of the same languages across different models was unattainable using the GAN model proposed by (Conneau, Lample, et al. 2017). In their approach, they tried to align different embedding models, such as GloVe, FastText, and Hyperwords-SVD, which were trained on the same English datasets. Despite these efforts, they were unable to achieve successful unsupervised English-to-English alignment. Subsequently, (Doval et al. 2019) shifted focus to aligning bilingual word embeddings using VecMap and MUSE algorithms, post-training on various corpora in different languages with a single word embedding model, FastText. They concluded that unsupervised alignment heavily depends on monolingual corpora and specific language pairs. Likewise, (Søgaard, Ruder, Vulić 2018) have carried out alignment processes between different languages pairs after training data on FastText model using skip-gram and CBOW methods. They concluded that eigen-vector similarities between monolingual corpora are associated with unsupervised alignment.

In our research on cross-model alignment, we examined three primary machine learning techniques: supervised, semi-supervised, and unsupervised learning. While supervised learning ensured alignment across different language pairs in cross-model scenarios, the cost of training data preparation steered our focus towards semi-supervised and unsupervised learning. The semi-supervised approach involved constructing a training vocabulary using common numerals in source and target

language pairs. However, we found that while effective for homogeneous models, this approach was inadequate for cross-model alignment. Consequently, we adopted two distinct domain adaptation techniques: CORAL and MMD.

Furthermore, (Jawanpuria, Meghwanshi, Mishra 2020) introduced a domain adaptation technique within models proposed by the same research group (MUSE), based on manifold alignment, an extension of the CORAL approach. Their results were compared with the Gromov-Wasserstein alignment of (Alvarez-Melis, Jaakkola 2018), an Optimal Transport approach, demonstrating the superiority of their manifold method on MUSE-provided word embeddings.

From an algorithmic perspective, we opted not to use the GAN approach implemented in the MUSE framework due to the instability of GAN training and the necessity for extensive hyperparameter searching to achieve alignment between different models. Additionally, we utilized an alignment algorithm by (Grave, Joulin, Berthet 2018) employing the Optimal Transport approach. However, we must note that we did not obtain successful results from alignment operations between different models. Therefore, our primary algorithm was the VecMap framework, which implements a variant of unsupervised ICP with a closed-form solution.

For all the aforementioned methods, it has been observed that alignment procedures are not effectively achievable with word embeddings generated by different models for different languages. This thesis investigates domain adaptation techniques that could potentially overcome this limitation, aiming to facilitate alignment between disparate models. In our cross-model alignment experiments, English served as a common pivot language. We attempted to make word vectors from GloVe_{en} compatible with FastText_{en} embeddings from various research groups using CORAL and MMD. Employing a pivot language for adaptation eased the challenge of unsupervised alignment between different languages.

The selection of CORAL and MMD as domain adaptation techniques represented two distinct approaches: CORAL focuses on adapting the feature space, whereas MMD, particularly through its Gaussian kernel, concentrates on sample space adaptation. This distinction underscores the differing philosophies in domain adaptation—features versus samples—and their significant implications for the alignment process. The success and applicability of each method vary depending on the nature of the datasets and the specific alignment challenges encountered.

Our experiments with CORAL revealed that the language pairs capable of alignment differed from those in experiments without domain adaptation, and we also observed variations in success rates. In contrast, the results from MMD experiments indicated

that despite minimizing discrepancy loss between two embedding spaces, successful alignment was not realized, necessitating further in-depth analysis.

Our experiments collectively indicate that there is no universal *silver bullet* domain adaptation technique effective in all scenarios. Furthermore, the size of the seed and the number of words used in the rotation process emerged as two critical parameters.

8.2 Cross Model Knowledge Distillation in Sentence Embeddings

This thesis presented alternative approaches for situations where traditional knowledge distillation is not feasible due to either inaccessible model parameters or the incompatibility of different deep learning frameworks. We have ventured into innovative methodologies that bypass these challenges, primarily focusing on improving the alignment of sentence vectors using FastText embeddings, diverging from the conventional practices associated with sBert sentence vectors and knowledge distillation methods.

Contrary to sBert, which leverages contrastive learning to group similar sentences and differentiate dissimilar ones, our method is founded on sentence vector training, but with a distinctive approach. We utilize vectors from varied architectures and apply loss functions centered on sentence similarities, moving away from sBert’s contrasting principle.

From the perspective of knowledge distillation, the standard approaches often focus on aligning hidden states or replicating logits between models. Our novel approach diverges by using the average of pre-logit hidden states, enabling the distillation process with a singular vector, rather than relying on the entire array of pre-logit hidden state vectors.

Our experimentation with dual combined loss functions showed varied results. The distance minimization strategy, which adapts FastText embeddings through an encoder model, unfortunately resulted in the loss of word content, leading to inferior sentence vector production. However, the distance & perplexity minimization model, employing a more sophisticated loss function combining cross-entropy and cosine similarity, demonstrated the significance of intricate loss functions and domain relevance in crafting models that generalize effectively across different linguistic scenarios. Additionally, our analysis indicated that datasets with greater domain coverage, meaning more vocabulary overlap with the training data, tend to perform better in classification tasks. This finding highlights the potential of our proposed models in computational linguistics.

A key focus of our research was on transferring information between different sentence embedding models. Here, an intriguing discovery was made: reducing the dimensions of the original sBert model does not adversely impact sentence evaluation tasks. This contradicts the prevalent belief that higher dimensions necessarily lead to improved performance, showing that a model with reduced dimensionality can yield comparable outcomes. This insight sheds new light on model efficiency and emphasizes the importance of dimensionality optimization in natural language processing tasks.



9

CONCLUSION

This thesis encompasses methods developed for aligning word embedding models trained on different datasets for the same or different languages using various models, and for aligning the representational powers of sentence embedding models that are trained on the same language but with different training datasets and models. These methods aim to address the overarching goal of making incompatible models compatible, enhancing their interoperability. In this chapter, we will highlight our key research findings, discuss the limitations encountered, and suggest potential future directions for both approaches.

9.1 Summary of Findings

In this section, we will briefly describe the main findings from our cross-model alignment experiments, encompassing both word and sentence embeddings. These findings represent the core achievements of our research, highlighting key insights and breakthroughs in the field of natural language processing.

9.1.1 Cross Model Word Embedding Alignment

The literature review reveals that all unsupervised word embedding alignment algorithms predominantly use orthogonal mapping. This method is a standard practice in the field due to its effectiveness in various settings. Aligning word embeddings typically use the most frequent words in both embedding spaces. However, this approach does not hold in cross-model alignments, indicating a limitation of this technique in more complex scenarios. Our experiments have identified that the size of the seed and the number of words used to determine the rotation matrix can affect the alignment process fundamentally. Unlike common assumptions, using a pivot language between different models for the same language does not necessarily aid in domain adaptation. This finding challenges current practices, suggests the need for alternative strategies in cross model alignment.

One of the intriguing observations is that the exact cause of misalignment in cross-model scenarios remains elusive. Understanding this aspect requires further in-depth investigation, as it is crucial for advancing the field. One of the intriguing observations is that the exact cause of misalignment in cross-model scenarios remains elusive. Understanding this aspect requires further in-depth investigation, as it is crucial for advancing the field.

9.1.2 Cross Model Knowledge Distillation in Sentence Embeddings

In our sentence embedding experiments, results obtained from the validation sets revealed that sentence similarity performance could be enhanced more easily than classification performance. We observed that classification performance could be improved by expanding the vocabulary in the training set. However, it's important to note that enhancing sentence representations invariably requires the use of an intermediate model.

Reflecting further on the validation phase of our research, we recognized significant potential for improvement in capturing the full spectrum of information from the sBert embedding space. This finding lays the groundwork for future research aimed at refining calibration techniques within representation spaces. One promising direction could be the expansion of training corpora size, which might further optimize the efficacy of these techniques.

9.2 Future Research Directions

In our research, we have specifically addressed challenges in both word and sentence model alignments. Our findings indicate that the lack of calibration between these models is a primary cause of inefficiency. Therefore, we conclude that achieving effective calibration between cross-models, whether in word or sentence embeddings, is crucial. This process is essential for ensuring consistency and compatibility across different models, which is vital for their practical application. Considering the diverse training data that might be used in the development of these models, aligning models created by distinct research groups or companies becomes highly significant. This alignment is not only imperative for NLP applications but also offers greater independence and flexibility to users in various fields. The ability to align word and sentence models developed under different conditions and by different entities can significantly enhance their applicability and utility.

Our ambition extends beyond resolving model alignment intricacies within and between word and sentence models. We aim to set new standards for model

interoperability, transcending traditional boundaries of model architectures. By overcoming these hurdles, we aspire to advance the field, unlocking the full potential of sophisticated word and sentence models for a broader range of applications.



REFERENCES

- Alaux, Jean et al. (2018). “Unsupervised hyperalignment for multilingual word embeddings”. In: *arXiv preprint arXiv:1811.01124*.
- Alvarez-Melis, David, Tommi S Jaakkola (2018). “Gromov-Wasserstein alignment of word embedding spaces”. In: *arXiv preprint arXiv:1809.00013*.
- Ammar, Waleed et al. (2016). “Massively multilingual word embeddings”. In: *arXiv preprint arXiv:1602.01925*.
- Arora, Sanjeev, Yingyu Liang, Tengyu Ma (2017). “A simple but tough-to-beat baseline for sentence embeddings”. In: *International conference on learning representations*.
- Artetxe, Mikel, Gorka Labaka, Eneko Agirre (Nov. 2016). “Learning principled bilingual mappings of word embeddings while preserving monolingual invariance”. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 2289–2294.
- (2017). “Learning bilingual word embeddings with (almost) no bilingual data”. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 451–462.
- (2018a). “A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings”. In: *arXiv preprint arXiv:1805.06297*.
- (2018b). “Generalizing and Improving Bilingual Word Embedding Mappings with a Multi-Step Framework of Linear Transformations.” In: *AAAI*, pp. 5012–5019.
- Besl, Paul J, Neil D McKay (1992). “Method for registration of 3-D shapes”. In: *Sensor fusion IV: control paradigms and data structures*. Vol. 1611. Spie, pp. 586–606.
- Bojanowski, Piotr, Armand Joulin (2017). *Unsupervised Learning by Predicting Noise*. arXiv: 1704.05310 [stat.ML].
- Cer, Daniel et al. (2018). *Universal Sentence Encoder*. arXiv: 1803.11175 [cs.CL].
- Conneau, Alexis, Douwe Kiela (2018). “SentEval: An Evaluation Toolkit for Universal Sentence Representations”. In: *CoRR abs/1803.05449*. arXiv: 1803.05449.
- Conneau, Alexis, Guillaume Lample, et al. (2017). “Word translation without parallel data”. In: *arXiv preprint arXiv:1710.04087*.
- Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv: 1810.04805 [cs.CL].
- Doval, Yera et al. (2019). “On the Robustness of Unsupervised and Semi-supervised Cross-lingual Word Embedding Learning”. In: *CoRR abs/1908.07742*. arXiv: 1908.07742. URL: <http://arxiv.org/abs/1908.07742>.
- Evangelidis, Georgios Dimitrios, Radu Horaud (2017). “Joint alignment of multiple point sets with batch and incremental expectation-maximization”. In: *IEEE transactions on pattern analysis and machine intelligence* 40.6, pp. 1397–1410.

- Faruqui, Manaal, Chris Dyer (2014). “Improving vector space word representations using multilingual correlation”. In: *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 462–471.
- Gao, Tianyu, Xingcheng Yao, Danqi Chen (Nov. 2021). “SimCSE: Simple Contrastive Learning of Sentence Embeddings”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 6894–6910. DOI: 10.18653/v1/2021.emnlp-main.552. URL: <https://aclanthology.org/2021.emnlp-main.552>.
- Grave, Edouard, Armand Joulin, Quentin Berthet (2018). “Unsupervised Alignment of Embeddings with Wasserstein Procrustes”. In: *arXiv preprint arXiv:1805.11222*.
- Gretton, Arthur et al. (2006). “A kernel method for the two-sample-problem”. In: *Advances in neural information processing systems* 19.
- Hardoon, David R, Sandor Szedmak, John Shawe-Taylor (2004). “Canonical correlation analysis: An overview with application to learning methods”. In: *Neural computation* 16.12, pp. 2639–2664.
- Hartmann, Mareike, Yova Kementchedjhieva, Anders Søgaard (2018). “Why is unsupervised alignment of English embeddings from different algorithms so hard?” In: *CoRR abs/1809.00150*. arXiv: 1809.00150. URL: <http://arxiv.org/abs/1809.00150>.
- Heyman, Geert et al. (2019). “Learning unsupervised multilingual word embeddings with incremental multilingual hubs”. 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)*, pp. 1890–1902.
- Hinton, Geoffrey, Oriol Vinyals, Jeff Dean (2015). *Distilling the Knowledge in a Neural Network*. arXiv: 1503.02531 [stat.ML].
- Jawanpuria, Pratik, Mayank Meghwanshi, Bamdev Mishra (2020). *Geometry-aware Domain Adaptation for Unsupervised Alignment of Word Embeddings*. arXiv: 2004.08243 [cs.LG].
- Kiros, Ryan et al. (2015). “Skip-Thought Vectors”. In: *CoRR abs/1506.06726*. arXiv: 1506.06726.
- Lan, Zhenzhong et al. (2020). *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. arXiv: 1909.11942 [cs.CL].
- Le, Quoc V, Tomas Mikolov (2014). *Distributed Representations of Sentences and Documents*. arXiv: 1405.4053 [cs.CL].
- Lian, Xin et al. (Jan. 28, 2020). “Unsupervised Multilingual Alignment using Wasserstein Barycenter”. In: *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2020*. arXiv: 2002.00743v2 [cs.CL].
- Liu, Yinhan et al. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. arXiv: 1907.11692 [cs.CL].
- Liu, Yufan et al. (2022). “Cross-Architecture Knowledge Distillation”. In: Publisher: arXiv Version Number: 2. (Visited on 04/10/2023).
- Logeswaran, Lajanugen, Honglak Lee (2018). *An efficient framework for learning sentence representations*. arXiv: 1803.02893 [cs.CL].
- Loshchilov, Ilya, Frank Hutter (2019). *Decoupled Weight Decay Regularization*. arXiv: 1711.05101 [cs.LG].

- Mao, Zhuoyuan, Tetsuji Nakagawa (2023). *LEALLA: Learning Lightweight Language-agnostic Sentence Embeddings with Knowledge Distillation*. arXiv: 2302.08387 [cs.CL].
- Meng, Xiao-Li, Donald B Rubin (1993). “Maximum likelihood estimation via the ECM algorithm: A general framework”. In: *Biometrika* 80.2, pp. 267–278.
- Mikolov, Tomas, Kai Chen, et al. (2013). “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781*.
- Mikolov, Tomas, Quoc V Le, Ilya Sutskever (2013). “Exploiting similarities among languages for machine translation”. In: *arXiv preprint arXiv:1309.4168*.
- Mikolov, Tomas, Ilya Sutskever, et al. (2013). “Distributed representations of words and phrases and their compositionality”. In: *Advances in neural information processing systems*, pp. 3111–3119.
- Pennington, Jeffrey, Richard Socher, Christopher D. Manning (2014). “GloVe: Global Vectors for Word Representation”. In: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.
- Radford, Alec, Karthik Narasimhan (2018). “Improving Language Understanding by Generative Pre-Training”. In: URL: <https://api.semanticscholar.org/CorpusID:49313245>.
- Reimers, Nils (2019). “Sentence Transformers: Multilingual Sentence, Paragraph, and Image Embeddings using BERT Co.” In: URL: <http://www.sbert.net>.
- Reimers, Nils, Iryna Gurevych (2019). “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *CoRR abs/1908.10084*. arXiv: 1908.10084.
- Sahlgren, Fredrik Carlsson Magnus (2021). *Sentence Embeddings by Ensemble Distillation*. arXiv: 2104.06719 [cs.CL].
- Sanh, Victor et al. (2020). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. arXiv: 1910.01108 [cs.CL].
- Søgaard, Anders, Sebastian Ruder, Ivan Vulić (2018). “On the limitations of unsupervised bilingual dictionary induction”. In: *arXiv preprint arXiv:1805.03620*.
- Sun, Baochen, Jiashi Feng, Kate Saenko (2016). “Return of frustratingly easy domain adaptation”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1.
- (2017). “Correlation alignment for unsupervised domain adaptation”. In: *Domain adaptation in computer vision applications*, pp. 153–171.
- Tang, Raphael et al. (2019). *Distilling Task-Specific Knowledge from BERT into Simple Neural Networks*. arXiv: 1903.12136 [cs.CL].
- Vaswani, Ashish et al. (2017). “Attention is all you need”. In: *Advances in neural information processing systems* 30.

A.1 Algorithms

Algorithm 1 Auxiliary Functions

```
1: function TOPKMEAN(Similarity :: Matrix, k :: Int = 10)
2:   return Top k mean neighbors
3: end function

4: function NORMALIZE(T :: Matrix)
5:   normalized  $\leftarrow$  unitEmbedding(T :: Matrix)
6:   normalized  $\leftarrow$  centerEmbedding(normalized)
7:   return normalized
8: end function

9: function SIMILARITIES(T :: Matrix)
10:  U, S, Vt = svd(T)
11:  return U * S * U' ▷ Cosine Similarity
12: end function

13: function UPDATEDICTIONARY(src :: Matrix, trg :: Matrix)
14:  simbwd = trg * src'
15:  simfwd = src * trg'
16:  knnbwd = TopKmean(simbwd, k=10)
17:  bestfwd = maximum(simfwd)
18:  simfwd = simfwd - (knnbwd / 2) ▷ CSLS calculation
19:  return getIndex(simfwd) ▷ return indices of best candidates
20: end function
```

Algorithm 2 Build Seed Dictionary

Require: X, Y : Source and target embedding matrices

Require: sim_size : Size of the similarity matrix, default 4000.

```
1: function BUILDSEEDDICTIONARY( $X :: Matrix, Y :: Matrix, sim\_size$ )
2:    $xsim \leftarrow \text{Similarities}(X[:, sim\_size])$ 
3:    $ysim \leftarrow \text{Similarities}(Y[:, sim\_size])$ 
4:    $\text{sort!}(ysim, dims = 1)$ 
5:    $\text{sort!}(xsim, dims = 1)$ 
6:    $xsim, ysim \leftarrow \text{map}(\text{Normalize}, [xsim, ysim])$ 
7:    $sim \leftarrow xsim^T * ysim$ 
8:    $sim \leftarrow \text{csls}(sim, k)$  ▷ CSLS calculation,  $k=10$ 
9:    $src\_idx \leftarrow \text{argmax}(sim, dims = 1)$  ▷ return best indices for source
10:   $trg\_idx \leftarrow \text{argmax}(sim, dims = 2)$  ▷ return best indices for target
11:  return  $src\_idx, trg\_idx$ 
12: end function
```

Algorithm 3 Unsupervised Word Mapping

Require: Source X and Target Y embeddings

```
1: function UNSUPERVISED_MAPPING( $X :: Matrix, Y :: Matrix$ )
2:   while not converged do
3:      $U, S, V = \text{svd}(Y[:, trg\_idx]' * X[:, src\_idx])$ 
4:      $W = V * U'$ 
5:      $XW = X * W$ 
6:      $trg\_idx = \text{UpdateDictionary}(XW, Y)$  ▷  $X \rightarrow Y$ 
7:      $src\_idx = \text{UpdateDictionary}(Y, XW)$  ▷  $Y \rightarrow X$ 
8:   end while
9:   return  $XW, Y$ 
10: end function
```

PUBLICATIONS FROM THE THESIS

Papers

Gunel, Kadir, Amasyali, M. Fatih (2023a). “*Boosting lightweight sentence embeddings with knowledge transfer from advanced models: A model-agnostic approach*”. In: *Applied Sciences*, 13(23):12586.

Conference Papers

Gunel, Kadir, Amasyali, M. Fatih (2023b). “*Model agnostic knowledge transfer methods for sentence embedding models*”. In: *2nd International Congress of Electrical and Computer Engineering (ICECENG)*.