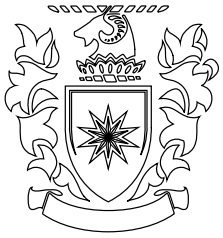


# CROSS-LINGUAL LEARNING IN LOW-RESOURCE

A THESIS PRESENTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE

**by**
JIAWEI ZHAO

**Supervisors:**
Dr. Andrew Gilman
A/Prof. Sean Zhu
Dr. Teo Susnjak

School of Natural and Computational Sciences
Massey University
Auckland, New Zealand

May 8, 2022

# Abstract

Current machine translation techniques were developed using predominantly rich resource language pairs. However, there is a broader range of languages used in practice around the world. For instance, machine translation between Finnish, Chinese and Russian is still not suitable for high-quality communication. This dissertation focuses on building cross-lingual models to address this issue. I aim to analyse the relationships between embeddings of different languages, especially low-resource languages.

I investigate four phenomena that can improve the translation of low-resource languages.

The first study concentrates on the non-linearity of cross-lingual word embeddings. Current approaches primarily focus on linear mapping between the word embeddings of different languages. However, those approaches don't seem to work as well with some language pairs, mostly if the two languages belong to different language families, e.g. English and Chinese. I hypothesise that linearity, which is often assumed in the geometric relationship between monolingual word embeddings of different languages, may not hold for all language pairs. I focus on investigating the relationship between word embeddings of languages in different language families. I show that non-linearity can better describe the relationship in those language pairs using multiple datasets.

The second study focuses on the unsupervised cross-lingual word embeddings for low-resource languages. Conventional approach to constructing cross-lingual word embeddings requires a large dictionary, which is hard to obtain for low-resource languages. I propose an unsupervised approach to learning cross-lingual word embeddings for low-resource languages. By incorporating kernel canonical correlation analysis, the proposed approach can better learn high-quality cross-lingual word embeddings in an unsupervised scenario.

The third study investigates a dictionary augmentation technique for low-resource languages.

A key challenge for constructing an accurately augmented dictionary is the high variance issue. I propose a semi-supervised method that can bootstrap a small dictionary into a larger high-quality dictionary.

The fourth study concentrates on the data insufficiency issue in speech translation. The lack of training data availability for low-resource languages limits the performance of end-to-end speech translation. I investigate the use of knowledge distillation to transfer knowledge from the machine translation task to the speech translation task and propose a new training methodology.

The results and analyses presented in this work show that a wide range of techniques can address issues that arise with low-resource languages in the machine translation field. This dissertation provides a deeper insight into understanding the word representations and structures in low-resource translation and should aid future researchers to better utilise their translation models.

# Ackonwledgement

My thanks go first to my committee, without whom this dissertation would not exist. To Andrew Gilman, my main advisor: I am glad you accepted me as your PhD student. This was the real start of my journey in Aotearoa and a fantastic time at Massey University. You have been a great supervisor. I want to thank you for all your guidance, encouragement and support. To Teo Susnjak: You have been so generous with your time, reading and commenting on my thesis. You helped me convey better what I meant to say. To Sean Zhu: I am grateful for the many priceless insights and your very concrete help in my work. Thank you for the advice and collaboration.

To my host during my visit to Euskal Herriko Unibertsitatea, Eneko Agirre, you played a significant role in my PhD and provided me with a deeper insight into my research.

To my mentors in Alibaba Group, Boxing Chen and Wei Luo: thank you for making me familiar with the priceless industrial knowledge.

I also want to thank my parents. They provided the best support they could during my research in Aotearoa. I would never have had this chance to be at this stage without them.

I also want to thank the friends I made at Massey, EHU and Alibaba. I would regret it, if I didn't write a 'thank you' for those who showed up in the dark time of my life.

This pandemic changed a lot of things in my life. My visit to Spain was just before the lockdown, and it took me one year to return to Aotearoa. Through the dangerous 2020 and 2021, I met many kind people: known, as well as, strangers. I would never have got through this without the happiness they gave to me. I believe this experience will be the most precious memory of my life in the coming future.

# Contents

**Contents** v

**List of Figures** x

**1 Introduction** 1

1.1 Machine Translation . . . . . 1

1.1.1 Rule-based Machine Translation . . . . . 1

1.1.2 Statistical Machine Translation . . . . . 2

1.2 A Brief History of Neural Machine Translation . . . . . 2

1.2.1 The Sequence-to-Sequence Model and the End-to-End NMT . . . 4

1.2.2 Encoder-Decoder Model with Attention . . . . . 5

1.2.3 Transformer . . . . . 8

1.3 Unsupervised Neural Machine Translation and Cross-lingual Word Embeddings . . . . . 11

1.3.1 Challenges with UNMT . . . . . 13

1.4 The End-to-End Speech Translation . . . . . 14

1.4.1 The Cascaded Model . . . . . 14

1.4.2 The End-to-End Model . . . . . 14

1.4.3 Challenges with End-to-End ST . . . . . 15

1.5 Contributions . . . . . 16

1.6 Dissertation Structure . . . . . 17

**2 Background and Theoretical Foundations** 19

2.1 Machine Learning . . . . . 19

2.2 Neural Networks . . . . . 20

2.2.1 Recurrent Neural Network . . . . . 22

2.2.2 Other Variants of Neural Networks . . . . . 22

2.3 Language Modelling . . . . . 23

2.3.1 The N-gram Model . . . 23
2.3.2 Neural Language Modelling . . . 23
2.4 Word Representation . . . 25
2.4.1 One-hot Vector Representation . . . 26
2.4.2 Distributed Word Representation . . . 26
2.4.3 Transformer-based Models . . . 29

**3 Non-Linearity in Cross-Lingual Word Embeddings 30**
3.1 Introduction . . . 30
3.2 Description of Notation . . . 31
3.3 Related Work: Mapping-based Approaches . . . 32
3.3.1 The Original Linear Mapping-based Method . . . 34
3.3.2 Orthogonal Methods . . . 34
3.3.3 The CCA-based Approach . . . 35
3.4 Description of the Mapping-based Cross-Lingual Word Embeddings Pipeline 37
3.4.1 Pre-processing . . . 37
3.4.2 Mapping . . . 38
3.4.3 Re-weighting . . . 38
3.4.4 Dimensionality Reduction . . . 39
3.5 Proposed Investigation: Non-Linear Methods in Cross-Lingual Word Embeddings . . . 39
3.5.1 KCCA . . . 40
3.5.2 KCCA-based Cross-lingual Word Embedding . . . 42
3.5.3 Deep Canonical Correlation Analysis . . . 43
3.5.4 DCCA-based Cross-lingual Word Embedding . . . 44
3.6 Experiments . . . 45
3.6.1 Datasets . . . 45
3.6.2 A New Word Translation Corpora: English-Chinese Dataset . . . 46
3.6.3 Evaluation . . . 46
3.6.4 Experimental Setup . . . 47
3.7 Results and Analysis . . . 48
3.7.1 Analysis Between Non-Linear Methods and Linear Methods . . . 48
3.7.2 Analysis Between Non-linear Methods . . . 49
3.7.3 Further Analysis for KCCA . . . 50

3.7.4 Ratio R . . . 54
3.8 Conclusion . . . 55

**4 Learning Unsupervised Cross-Lingual Word Embeddings with Non-Linear Mapping 60**
4.1 Introduction . . . 60
4.2 Related Works . . . 61
4.2.1 Supervised Cross-Lingual Word Embedding Approaches . . . 61
4.2.2 Unsupervised Mapping-based Cross-Lingual Word Embedding Approaches . . . 62
4.2.3 Self-Learning . . . 63
4.3 Pipeline . . . 64
4.3.1 Unsupervised Dictionary Initialisation . . . 65
4.3.2 Embedding Mapping . . . 66
4.3.3 Dictionary Induction . . . 66
4.4 Proposed Investigation: Learning Unsupervised Cross-lingual Word Embedding with Non-linear Mapping . . . 67
4.4.1 Step One: Unsupervised Dictionary Initialisation: . . . 68
4.4.2 Step Two: Self-learning . . . 68
4.4.3 Step Three: Final Projection . . . 70
4.4.4 Convergence Criterion . . . 71
4.5 Experiments . . . 71
4.5.1 Dataset . . . 71
4.5.2 Experiment Details . . . 72
4.5.3 Evaluation . . . 72
4.6 Result Analysis . . . 73
4.6.1 CCA-based Self-learning Iterations . . . 73
4.6.2 Final Projection Step . . . 74
4.6.3 Comparison with State-of-the-art . . . 77
4.7 Conclusion & Future Works . . . 77

**5 Consistency-based Cross-Lingual Word Embeddings 79**
5.1 Introduction . . . 80
5.1.1 Bias-Variance Tradeoff in the Mapping-based Methods . . . 80

5.1.2 The Ensemble Model . . . 81
5.2 Bagging . . . 82
5.2.1 Cross-Consistency . . . 85
5.2.2 Self-Consistency . . . 86
5.3 Consistency-based Cross-Lingual Word Embeddings . . . 88
5.4 Experiments . . . 93
5.4.1 Dataset . . . 93
5.4.2 Experimental Details . . . 94
5.4.3 Ensemble Model . . . 95
5.5 Result Analysis . . . 96
5.5.1 Comparing to the Pipeline of Mapping-based Method . . . 96
5.5.2 Compare to Naive Bagging . . . 97
5.5.3 Analysis of the RANSAC Initialisation . . . 97
5.5.4 Analysis of the Mutual Nearest Neighbours Dictionary Induction . 98
5.5.5 Analysis on the Frequency-based Dictionary Cutoff Strategy . . . 99
5.5.6 Model Number's Impact . . . 99
5.5.7 Analysis of the Ensemble Strategy . . . 100
5.6 Conclusion & Future work . . . 101

**6 Mutual Learning for Speech Translation 103**
6.1 Introduction . . . 104
6.2 Background . . . 105
6.2.1 Definition of Speech Translation . . . 105
6.2.2 Cascaded Model . . . 105
6.2.3 The Vanilla Model of End-End Speech Translation . . . 106
6.2.4 The Knowledge Distillation for End-to-End Speech Translation . . 107
6.3 Proposed Mutual-Learning-based Speech Translation . . . 108
6.3.1 Model Description . . . 108
6.3.2 Training Strategy . . . 109
6.4 Experiments . . . 110
6.4.1 Dataset . . . 110
6.4.2 Training Details . . . 111
6.5 Results and Analysis . . . 111
6.5.1 Cascaded Model Comparison . . . 111

6.5.2 Knowledge Distillation Model Comparison . . . . . 112
6.5.3 Multi-Task Learning Model Comparison . . . . . 113
6.5.4 Performance of the MT System . . . . . 113
6.5.5 Analysis of the Cycling Annealing Schedule . . . . . 114
6.6 Conclusion & Future Works . . . . . 114

**7 Conclusion** **116**

**Bibliography** **120**

# List of Figures


1 SMT model. The translation model and language model is trained by the parallel corpora, and then the two models are used to decode a source sentence $s$ into its target translation sentence $\hat{t}$. . . . . 3

2 A brief timeline of neural machine translation with key milestones. All of the models are encoder-decoder-based-models. . . . . 4

3 An example of sequence to sequence model for MT. The input word sequence is initially encoded by the Encoder RNN to an intermediate representation $h$, then the decoder RNN decodes $h$ into the output sequence $o1, ....$ The model is trained by minimising the distance between $o1, ...$ and $y1, ...$ using a criterion like maximum likelihood estimation (MLE). . . . . 6

4 The architecture of attention-based encoder-decoder model. In each time step $t$ of decoder RNN, the hidden state $s_t$ is calculated using Equation 1.5. Each output $O_t$ is supervised by the corresponding target token $y_t$. . . . . 7

5 Transformer architecture. . . . . 8

6 An example of a scaled dot-product attention. . . . . 10

7 A multi-head attention is a concatenation of multiple scaled dot-product attentions. . . . . 10

8 Unsupervised neural machine translation model. . . . . 11

9 Back-translation process. . . . . 13

10 The end-to-end speech translation research area. . . . . 15

11 A neuron of a neural network. . . . . 20

12 The structure of a fully-connected feed-forward neural network. . . . . 21

13 The structure of a recurrent neural network. . . . . 22

14 Neural language model. The context words with one-hot vector form are first mapped to word embedding vectors through a look-up table, then the word embeddings are used as input to predict the next word. . . . . 24

15 Training process of the RNN neural language model at time step four with an input sentence 'I like to play football'. The input of the RNN is one-hot vector representation of words in the input sequence. The target $y$ is also a one hot representation. In each time step, RNN calculates the probability distribution of the next word. . . . . 25
16 Skip-gram model and CBOW model. . . . . 28
17 A visual representation of the concepts used in this chapter. $V_s$ is the vocabulary of the source corpus. The $\tilde{X}_s$ is the word embedding matrix of the vocabulary. Each row of word embedding matrix $\tilde{X}_s$ is a word vector representing a word in the vocabulary. Given a bilingual dictionary $D$, the source words of the dictionary are used to formulate a word embedding matrix $X_s \in \tilde{X}_s$. Each row of $X_s$ is a word embedding representing a source language word from the dictionary. . . . . 32
18 mapping-based-apporach . . . . . 33
19 Visualisation of the cross-lingual word embeddings construction process using mapping-based approaches. . . . . 38
20 DCCA-based cross-lingual word embeddings. . . . . 44
21 Word translation accuracy of test set dictionary versus different values of gamma on En-De and En-Es dataset. x-axis is gamma, and y-axis is word translation accuracy of the test. . . . . 50
22 Word translation accuracy of test set dictionary versus different values of gamma on En-It and En-Fi dataset. x-axis is gamma, and y-axis is word translation accuracy of the test. . . . . 51
23 Word translation accuracy of En-It and En-De test set dictionary versus different values of weight $w$. x-axis is the weight parameter, and y-axis is word translation accuracy of the test. . . . . 52
24 Word translation accuracy of En-Es and En-Fi test set dictionary versus different values of weight $w$. x-axis is the weight parameter, and y-axis is word translation accuracy of the test. . . . . 53
25 Word translation accuracy of test set of En-Es dataset applying different dimensions. . . . . 54
26 Word translation accuracy comparison of CCA and KCCA when they have the same dimension. The blue line is CCA and the red line represents KCCA. 54

27 The translation result on the En-It and En-De test sets. $x$ axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only. . . . . 56

28 The translation result on the En-Es and En-Fi test sets. $x$ axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only. . . . . 57

29 The translation result on the En-Zh test sets. The $x$ axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only. . . . . 58

30 CCA-based self-learning scenario. The seed dictionary $D$ is separated as two matrices of word embeddings $x$ and $y$. CCA is applied on $x$ and $y$ seeks to find two projection matrices $w_x$ and $w_y$. Then the whole vocabulary is projected into the new feature space. Nearest neighbours are used to create a new dictionary and the whole operation is repeated. . . . . 64

31 Word translation accuracy of unsupervised CCA-Based cross-lingual word embedding on the test set of En-Es dataset. . . . . 70

32 Word translation accuracy of test set during iterations on En-It dataset. Using CCA-based mapping in self-learning process. The final step mapping is KCCA mapping. . . . . 73

33 Word translation accuracy of test set during iterations on En-De dataset. Using CCA-based mapping in self-learning process. The final step mapping is KCCA mapping. . . . . 74
34 Word translation accuracy of test set during iterations on En-It dataset. The CCA-based mapping approach is used in self-learning process. The final step mapping is orthogonal mapping. . . . . 75
35 Word translation accuracy of test set during iterations on En-De dataset. The CCA-based mapping approach is used in self-learning process. The final step mapping is orthogonal mapping. . . . . 75
36 Word translation accuracy of test set during iterations on En-De, En-Es, En-It dataset. . . . . 76
37 Word translation accuracy of test set on En-Fi dataset. The proposed system spend much more time on finding the first reasonable local optima. . . . . 76

38 A brief summary of the dictionary augmentation models. . . . . 80
39 An ensemble-like dictionary augmentation model. . . . . 82
40 Consistent word pairs of test sets in a different number of models. The y-axes represent how many models are used. The x-axes represent the number of word pairs. The blue bars are the consistent pairs and the orange bars represent the correctly translated word pairs that are in the consistent pairs. . . 83
41 The cross-consistency on En-Zh and En-Es dataset. The figures show the result of two sets of parameters of KCCA- and SVD- based mapping. . . . . 87
42 Self-consistency on the En-Zh dataset. '1' and '2' means two training subsets. 89
43 Self-consistency of different mapping-based methods on the En-Es dataset. '1' and '2' means two training subsets. . . . . 90
44 An example of my proposed consistency-based model between English and German words. . . . . 91
45 An illustration of the model agreement process. . . . . 92
46 The accuracy evaluated on training set using the learned dictionary. . . . . 98
47 The word translation accuracy of training word pairs with a different number of models. . . . . 100
48 Number of the correctly translated word pairs within consistent word pairs when different number the models are combined. . . . . 101

49 Cascaded model. . . . . 106

50 The end-to-end speech translation model. . . . . 107
51 The proposed deep mutual learning scenario. The training objective contains four separate components, the reconstruction losses of ST and MT ($LC_{st}$ and $LC_{mt}$) and KL divergence between outputs of ST and MT ($KL_1$ and $KL_2$). . . 109
52 Mutual-Learning model detail. The structure of my mutual learning paradigm is two stacked transformer. . . . . 112
53 The loss for cycling annealing schedule on En-Fr dataset. . . . . 115

CHAPTER 1

# Introduction

## 1.1 Machine Translation

Machine translation (MT) is a sub-field of natural language processing (NLP). It aims to automatically translate language or speech from one to another without any manual involvement (Yang et al., 2020a). The attainment of accurate machine translation has been one of the primary objectives within the NLP field. It paves a path to fully establishing the understanding of human language by artificial intelligence systems.

Machine translation has been under investigation since 1950s. Until 2010s, rule-based machine translation (RBST) and statistical machine translation (SMT) dominated the field (Forcada et al., 2011; Koehn et al., 2003). In recent years, neural machine translation (NMT) has become the primary research direction within the MT field (Cho et al., 2014a).

### 1.1.1 Rule-based Machine Translation

Rule-based machine translation (RBST) was one of the initial approaches for MT. The hypothesis of RBST is that most words in one language can find their corresponding translated word (word with the same meaning) in another language. Based on this hypothesis, the translation process of RBST can be seen as a replacement process: given a sentence in one language, the translation could be achieved through a direct word-by-word translation. Instead of directly translating a sentence word by word, the RBMT additionally considers the syntax rules of different languages. For instance:

The lady has a blue hat.

La dama tiene un sombrero azul.

These sentences are translated decently by considering the word syntax rules of two languages. However, the weakness of RBST lies in its robustness. A translated sentence with a correct syntax may not have a reasonable meaning. A famous example is given by Bar-Hillel (1960) [1]:

*'The pen is in the box'*

*'The box is in the pen'*

There is no difference in the syntax structure of those two sentences, but the first sentence is more reasonable than the second one. This is because the word 'pen' is a polysemant and has multiple meanings, one of them also being 'fence' in English. The inability of computers to understand context, and thus have the ability to consistently make reasonable translations using RBST, has rendered this approach insufficient.

### 1.1.2 Statistical Machine Translation

Statistical machine translation has played a significant role in MT over several decades. It has been used widely and successfully across industry and real-world applications. SMT aims to build a statistical model based on two parallel corpora. Given a source sentence $S$ and a target sentence $T$, the SMT model aims to maximise the probability of the correct translation from $S$ to $T$:

$$\underset{T}{\operatorname{argmax}}\, P(T|S) \tag{1.1}$$

According to Bayes' rule, Equation 1.1 can be reformulated into:

$$\underset{T}{\operatorname{argmax}}\, P(S|T)P(S) \tag{1.2}$$

In Equation 1.2, $P(S)$ refers to the probability of a source sentence. In SMT, $P(S)$ refers to the "language model", which denotes the probability of a known (or fluent) source sentence. The $P(S|T)$ is the "translation model" in SMT, which indicates the probability of the translation from $S$ to $T$.

## 1.2 A Brief History of Neural Machine Translation

The Neural Machine Translation (NMT) model is the most popular model for MT. NMT model provides robust and reliable translation results that are now at a level of quality

[1] Yang et al. (2020a) claims that this example is provided by Marvin Minskey in 1966

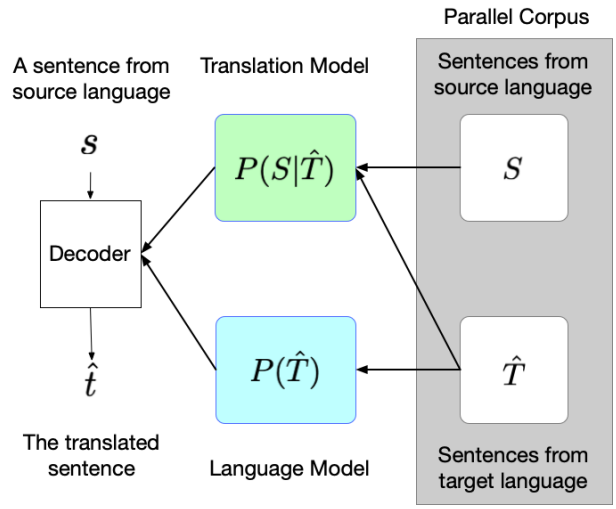



**Figure 1:** SMT model. The translation model and language model is trained by the parallel corpora, and then the two models are used to decode a source sentence s into its target translation sentence $\hat{t}$.

sufficient enough to impact people's everyday lives.

Earlier NMT models were encoder-decoder architectures based on recurrent neural network (RNN). In this model, an RNN encoder encodes the input sentence into an intermediate representation, and the decoder decodes this intermediate representation to the target translation sentence (Cho et al., 2014a; Sutskever et al., 2014). The word embeddings were first used in NMT models to represent words (Sutskever et al., 2014).

However, the key constraint of RNN-based models is their limitation at extracting long dependencies of long sentences. This is because an RNN struggles to deal with the vanishing gradient problem – when the number of time steps of an RNN increases, the gradient becomes proportionally smaller, effectively preventing the weights update. Thus, the long sentence translation had become a bottleneck in RNN-based NMT models. The attention-based model is proposed to address the bottleneck. An attention-based model adds input sequences' positional information during its training process, helping the model extract the sentence structure and long dependencies (Bahdanau et al., 2015).

Due to the success of the attention models, an increasing number of real-world translation applications have touched our daily lives. In order to be effective, NMT modes require a large amount of data to obtain decent translation results, so a large and parallel structure is required to model the data. However, there is another limitation of RNN: The inner structure of RNN does not support parallel computing, which makes the computational cost of the RNN-based models expensive.

Nowadays, lower computational costs and faster computational speeds have become more and more critical for NMT. Therefore, RNN-based models are not ideal for NMT

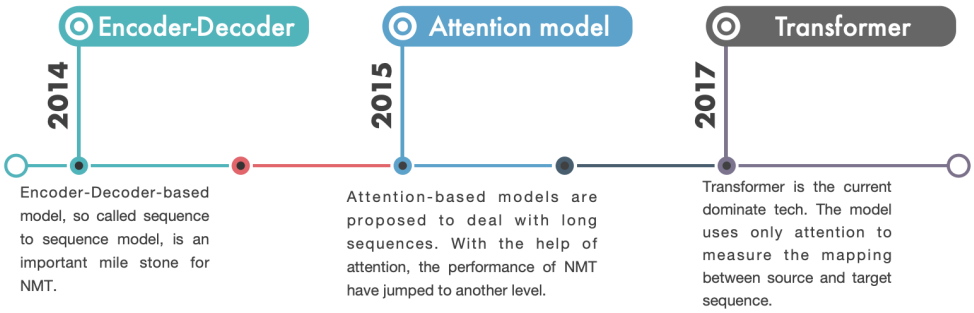



**Figure 2:** A brief timeline of neural machine translation with key milestones. All of the models are encoder-decoder-based-models.

systems. The Convolutional Neural Network (CNN)-based NMT models have been proposed to mitigate the issue in RNN-based NMT structures (Gehring et al., 2017). There are two advantages for modelling sentences using CNN: Firstly, the inner structure of CNN is more suitable for parallel computing. Secondly, the CNN can be extended to much larger input sizes to capture the long dependencies of sentences better.

Hence, CNN-based model has become popular in modelling sentences. Kalchbrenner et al. (2016) used CNN to model sentences without any attention. (Bradbury et al., 2016) tackle the issue for RNN by combining RNN and CNN together. (Gehring et al., 2017) introduce attention in CNN-based NMT models to better extract the position information in long sentences.

More recently, the *transformer* model has dominated the NMT models. Those models are widely applied in a commercial setting. The transformer-based models depend only on the attention mechanism (Vaswani et al., 2017), which offers the advantages of both CNN- and RNN-based structures. Significantly, the transformer can handle a large amount of data and effectively extract information from large corpora. Figure 2 shows the brief historical progression and significant milestones in the evolution of NMT.

### 1.2.1 The Sequence-to-Sequence Model and the End-to-End NMT

The translation methods within MT can take different inputs and this is one of the key differentiating factors between them. Some can take as inputs the entire document, a paragraph or individual sentences.

End-to-end neural machine translation aims to directly learn the mapping between the source sentence and the target sentence via neural networks. Given the success of this approach, NMT models rapidly superseded other conventional approaches in MT (Bahdanau

et al., 2015; Kalchbrenner and Blunsom, 2013). This work focuses on sentence-level machine translation. Thus, the input of the NMT model is a sentence and the output is a sentence. These NMT models are referred to as *sequence-to-sequence* models.

An important earlier work in modelling sentences is proposed by Cho et al. (2014b). This work used the recurrent neural network (RNN) to model the time sequence information. The encoder RNN encodes the input word sequence into a fixed length representation, and another decoder RNN, decodes this representation (which is the last hidden state of the encoder RNN) into the target sequence. However, this approach suffered from a weakness whereby RNN can not model long sequences with the effect of making it unreliable for translating long sentences (Sutskever et al., 2014). Sutskever et al. (2014) extended the initial approach to RNN with Long-Short-Term-Memory (LSTM) units, which improved the ability to handle long sequence information.

In their work, given a sentence $X = (x_1, x_2, ..., x_T)$, the encoder RNN was used to estimate the probability of $P(y_1, y_2, ..., y_T | x_2, ..., x_S)$. Where $Y' = (y_1, ..., y'_T)$ is the output of the RNN. During training, the sequence model first receives the last hidden state $h$ of the encoder RNN and uses $h$ to compute the conditional probability of $P(y_1, y_2, ..., y_T | x_2, ..., x_S)$ with a following language modelling process:

$$P(y_1, y_2, ..., y_T | x_2, ..., x_S) = \prod_{t=1}^{T} P(y_t | y_1, ..., y_{t-1}, h) \tag{1.3}$$

In Equation 1.3, each component $P(y_t)$ is modelled by a unit of RNN (or the LSTM-based RNN). A significant feature of this work is the modelling process. Each sentence must end with a same symbol <EOS>, which enables the sequence-to-sequence-model to recognise the length of the sentence. For example, a sentence 'The price of this book <EOS>' is first encoded by the encoder RNN, and then the decoder RNN decodes this into 'El precio de este libro <EOS>'. This method was found to be effective at translation performance by Sutskever et al. (2014) and is illustrated in Figure 3.

### 1.2.2 Encoder-Decoder Model with Attention

RNN-based models suffer a limitation from the length of the sequence because the encoder-decoder-based models compress the whole input sequence into a simple representation which is the last hidden state of the encoder RNN. Therefore, the performance for translating long sentences has been an issue for sequence to sequence models. Bahdanau et al. (2015) extend the traditional RNN-based approach to a new sequence-to-sequence struc-

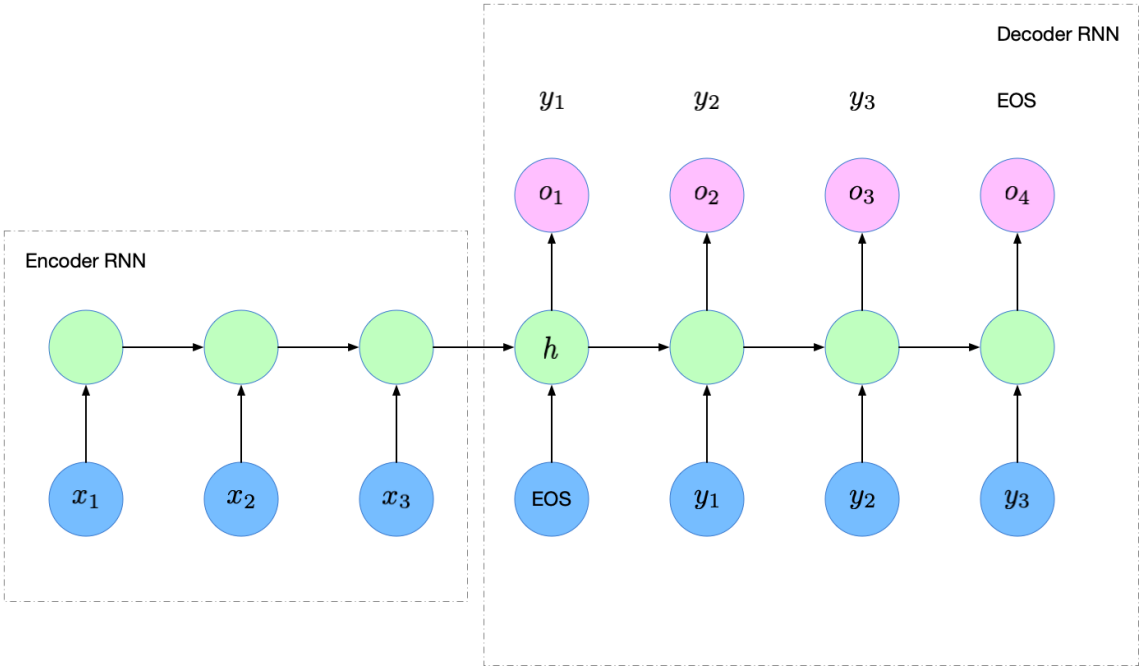



**Figure 3:** An example of sequence to sequence model for MT. The input word sequence is initially encoded by the Encoder RNN to an intermediate representation $h$, then the decoder RNN decodes $h$ into the output sequence $o1, ....$ The model is trained by minimising the distance between $o1, ...$ and $y1, ...$ using a criterion like maximum likelihood estimation (MLE).

ture called the *Attention-based* model. Now, instead of using only the last hidden state, each hidden state $h_i$ of the encoder is used. An attention $c_t$ is calculated when decoding as follows:

$$p(y_t|y_1, ..., y_{t-1}) = f(y_{t-1}, s_t, c_t) \tag{1.4}$$

where $s_t$ is the corresponding hidden state of decoder RNN. The $s_t$ is calculated by Equation 1.5:

$$s_t = f(s_{t-1}, y_{t-1}, c_t) \tag{1.5}$$

The attention $c_t$ is a weighted distribution of the hidden states of the encoder defined as:

$$c_t = \sum_{s=1}^{S} a_{st} h_t \tag{1.6}$$

where $a_{st}$ are the weight parameters of the each input token of the source sentence. The weight $a_{st}$ is a measure of normalised similarity between the current decoding time step and each encoding time step:

$$a_{st} = \frac{exp(e_{st})}{\sum_{k=1}^{S} e_{sk}} \tag{1.7}$$

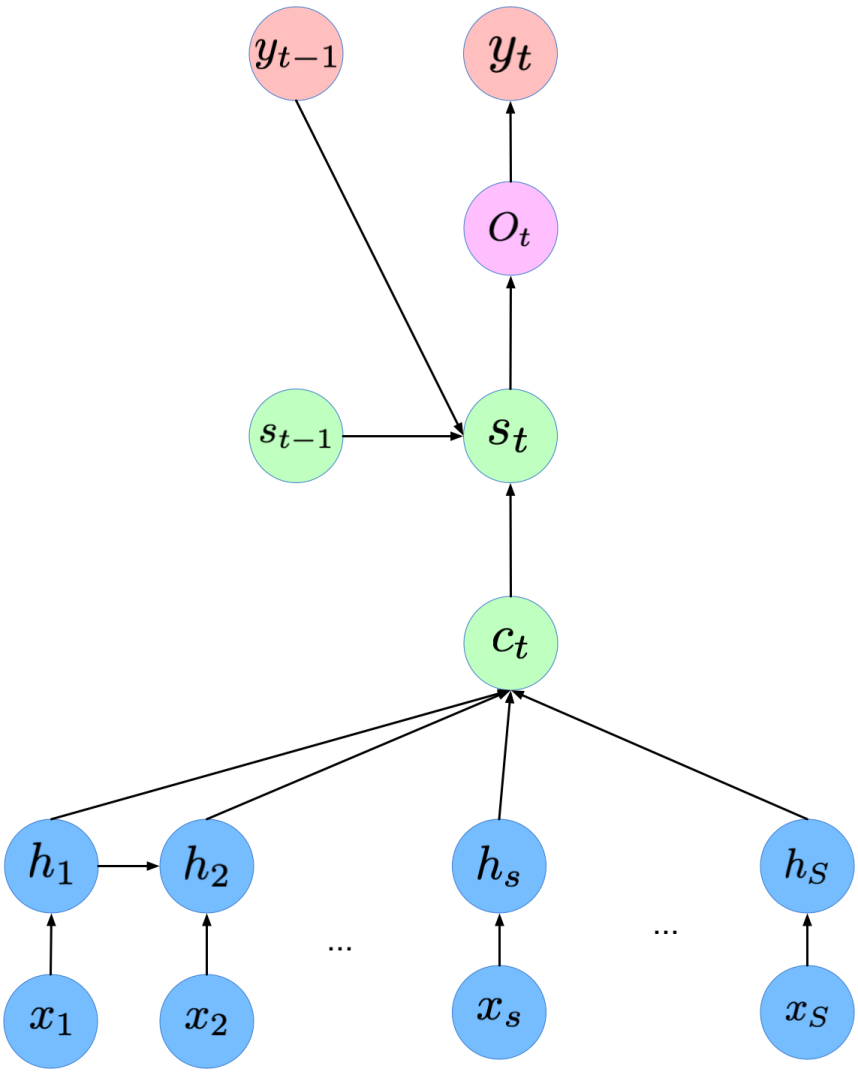



**Figure 4:** The architecture of attention-based encoder-decoder model. In each time step $t$ of decoder RNN, the hidden state $s_t$ is calculated using Equation 1.5. Each output $O_t$ is supervised by the corresponding target token $y_t$.

where $e_{st}$ is an similarity measure between the similarity of the hidden state of time step $t$ and the hidden state of encoder's time step $s$. Bahdanau et al. (2015) parameterise the $e_{st}$ as a neural network $\lambda$ to measure this similarity as:

$$e_{st} = \lambda(h_s, h_t) \tag{1.8}$$

Equation 1.8 is also called the *alignment model* which measures the condition between the source input and the target. Figure 4 summarises the attention model. With the attention paradigm, each time step in the decoding process is additionally evaluated by a weighted sum of the encoder inputs, which reflects the importance of each input to the current output.

In addition, there are two types of attentions, *soft attention* and *hard attention* described

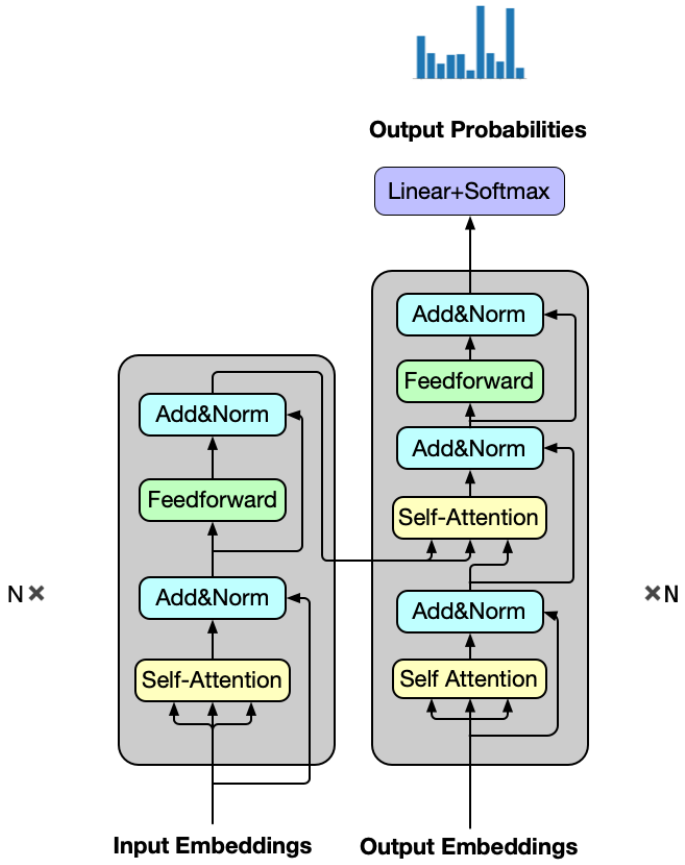



**Figure 5:** Transformer architecture.

by Luong et al. (2015). The hard attention is the attention described before. The soft attention method only uses small groups of positions in the encoder, instead of using all of the encoder hidden layers to calculate the attention.

### 1.2.3 Transformer

Transformer, proposed by Vaswani et al. (2017), is an essential technique for NMT use nowadays in both academia and industry. The transformer is appealing because of its simplicity and efficiency compared to the traditional RNN-based models.

The high computational cost limits the traditional RNN-based models. This limitation becomes a critical shortcoming when the data size increases, making it hard to deploy the RNN-based model into a parallel training strategy like CNN. Inspired by the attention-based models, Vaswani et al. (2017) propose the Transformer model, which eliminates the recurrent model and instead is only based on attention mechanisms to describe the mapping between the source and target sentences.

As illustrated in Figure 5, transformer is still an encoder-decoder model. The encoder encodes the input embeddings into some intermediate representation and the decoder decodes it into the target language. The model is connected with normalisation layers and

attention layers. The addition of attention in the transformer mechanism has also been recently proposed and termed as *multi-head self-attention mechanism* or *self attention* for simplicity.

In this approach, the encoder is an N stack of identical layers. For each layer, there are two sub-layers. One sub-layer is the self attention, and the other sub-layer is a simple feed-forward neural network. Each sub-layer has a residual connection (He et al., 2016). The residual connection can be represented as:

$$\mathtt{output} = x + \mathtt{sublayer}(x) \tag{1.9}$$

The above means that the output of the layer is the sum of the sub-layer x's output and the input itself. Each sub-layer is fully connected with a normalisation layer. The decoder consists of a N stack of identical blocks. Each block consist of three sub-layers. One sub-layer is a self-attention layer, another is a multi-head attention layer, while the remaining one is a feed-forward neural network. Similar to the encoder, each sub-layer is fully connected with a normalisation layer and residual connection is applied.

The self-attention is a concatenation of multiple *scaled-dot-product-attentions*. The input of *scaled-dot-product-attentions* consists of queries Q, keys K, and values V. Given this input $< Q, K, V >$, this attention is calculated by:

$$\mathtt{attention}\,(Q, K, V) = \mathtt{softmax}\left(\frac{QK^{\intercal} + M}{\sqrt{d_k}}V\right) \tag{1.10}$$

where d is the dimension of the key value K. For translation tasks, Q and K are identical, which are input tokens. The V are the corresponding target tokens. The mask matrix M is the padding mask which avoids the sequence pad information involved in the training procedure. Figure 6 illustrates the scaled dot-product attention. If we concatenate multiple attentions, we use Equation 1.10, from which the output is the multi-head attention described in Figure 7.

With the nature of parallel modelling, transformer trains much faster than traditional RNN-based sequence models. Additionally, and surprisingly, the performance of the transformer is much better than previous modelling approaches. Therefore, transformer is widely used in many areas. In Chapter 6, this model is applied in conjunction with my proposed method in this study.

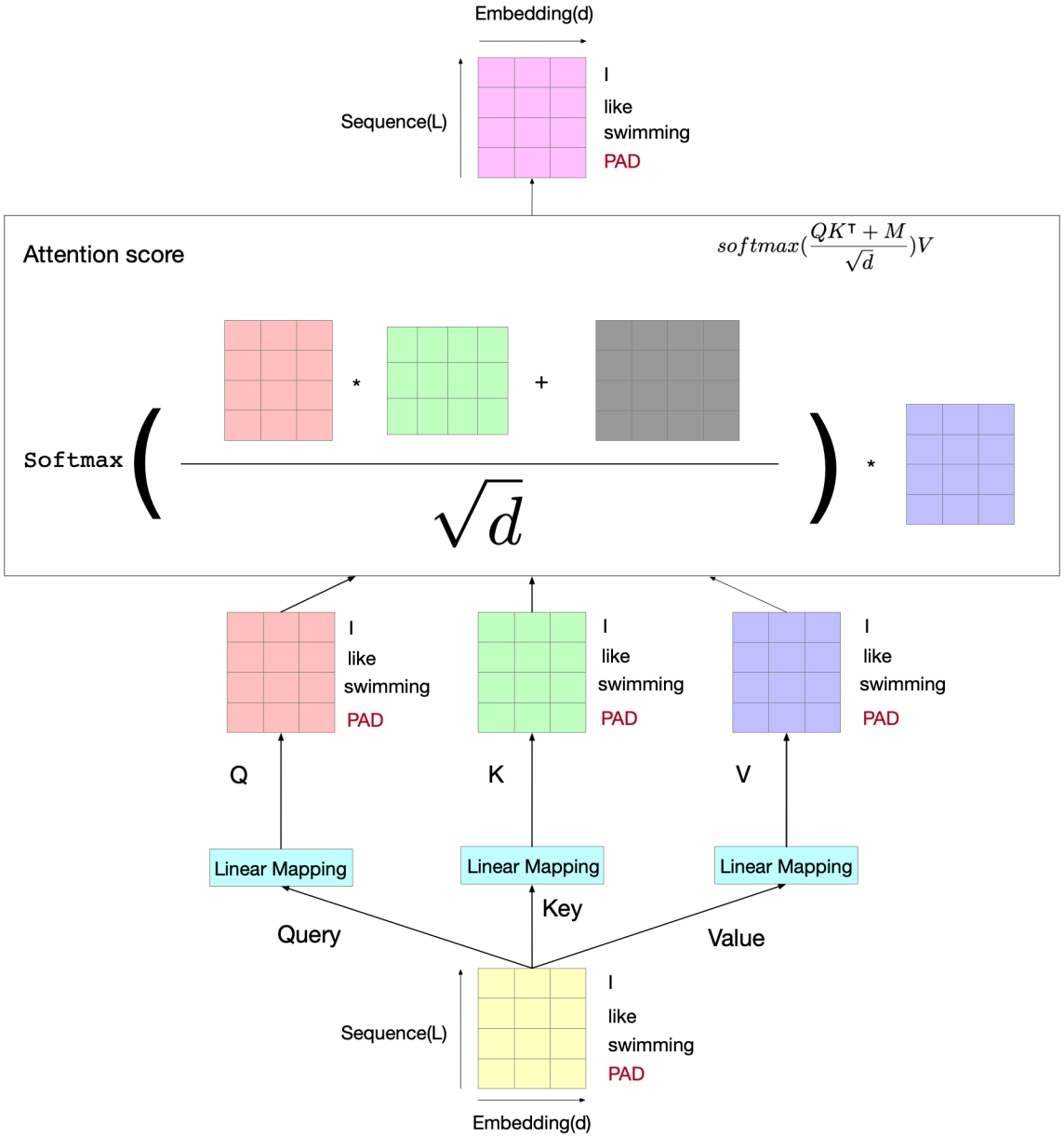



**Figure 6:** An example of a scaled dot-product attention.

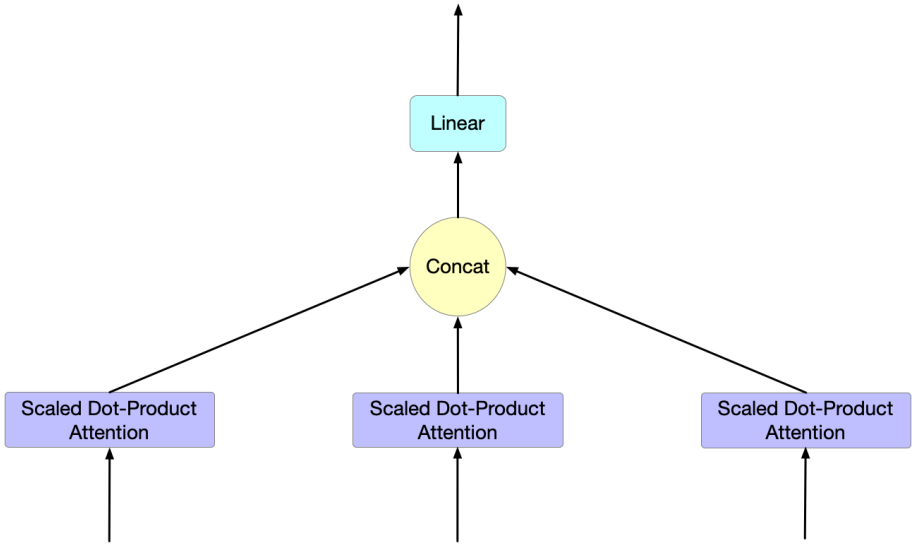



**Figure 7:** A multi-head attention is a concatenation of multiple scaled dot-product attentions.

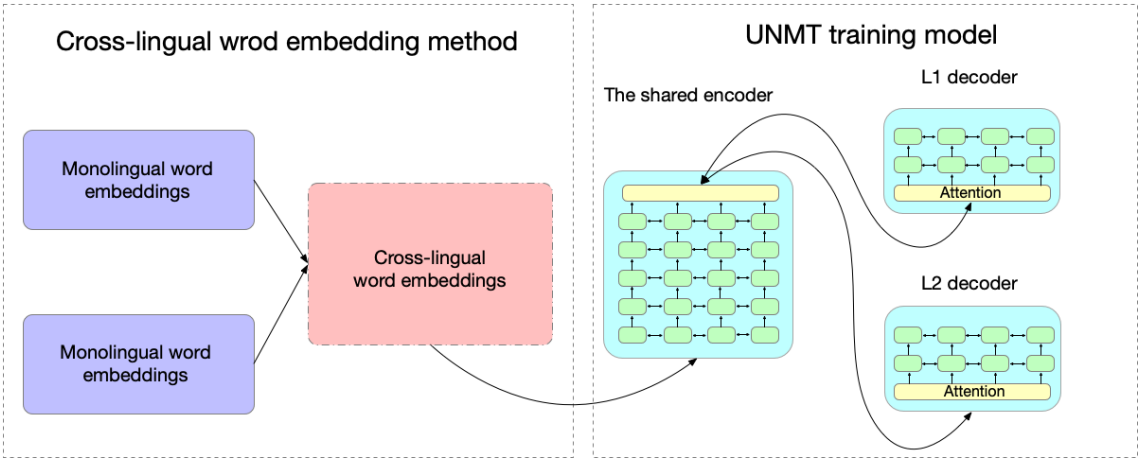



**Figure 8:** Unsupervised neural machine translation model.

## 1.3 Unsupervised Neural Machine Translation and Cross-lingual Word Embeddings

A critical bottleneck of NMT in recent years has been data insufficiency. This has especially been the case for low-resource language pairs when the parallel data is hard to obtain. Therefore, some research efforts have focused on mitigating this issue. One particular research direction for tackling this challenge is called unsupervised neural machine translation (UNMT). UNMT aims to translate languages based solely on monolingual corpora. Artetxe et al. (2018c) demonstrates one of the earliest attempts to implement UNMT showing that a reasonable translation performance is possible.

UNMT uses an essential technique called *cross-lingual word embedding methods*, which maps monolingual word embeddings into a shared feature space. The obtained word representation is cross-lingual word embedding that enables knowledge transfer to take place via the shared feature space and significantly helps UNMT provide competitive translation results. Section 3.1 introduces this approach in detail. As shown in Figure 8, UNMT has one shared encoder and two separate decoders. Each pair of encoder and decoder is an RNN Encoder-Decoder model with attention, which I describe in greater detail in Section 1.2.2. Instead of word sequences, the input of this model are fixed-size cross-lingual word embeddings.

The training process consists of two parts: *denoised auto-encoding* and *back translation*. The denoised auto-encoding process feeds the system with noisy input and tries to reconstruct the original sentence. For instance, given a sentence from language L1, the model first shuffles the input sentence, and then attempts to reconstruct the sentence into the original order using the shared encoder and L1 decoder. There are many noise-

inducing strategies such as re-ordering of the words from input sentence; word deletions or words-swapping. During the training process, all of those strategies are applied randomly onto the input sentences.

Back-translation is a data augmentation method dealing with language pairs in the presence of inadequate data (Sennrich et al., 2016). In an UNMT model, the back-translation serves as a training task, which can be divided into three steps:

1. The inference mode of the shared encoder and L2 decoder is used to translate a source sentence into a pseudo translation. The source sentence and the generated pseudo translation in target language together are called *pseudo-translated* sentence pairs.

2. The pseudo translation is feed into the shared encoder and L2 decoder in training mode, aiming to translate it into the source sentence.

3. A new language pair is generated: The pseudo translation and the source sentence. This sentence pair can then be further used to train the shared encoder and L1 decoder system.

During the training process of UNMT, a source sentence S from the language L1 is first translated into a target sentence $\hat{\mathrm{T}}$ in the language L2 using the inference mode of the shared encoder and L2 decoder, then the shared encoder and L1 decoder are used to translate $\hat{\mathrm{T}}$ to S. Figure 9 illustrates this process.

During the training process, the mini-batches of sentence data are used to train two objectives described before in an alternative fashion: de-noising and back translation. The training process of UNMT can be separated into four steps:

1. The L1 de-noising. The sentence mini-batches of L1 language is used to train the shared encoder and L1 decoder with the de-noising task I described before.

2. The L2 de-noising.The sentence mini-batches of L2 language is used to train the shared encoder and L1 decoder with the de-noising task.

3. Back-translate the sentences from L1 to L2.

4. Back-translate the sentences from L2 to L1.

Those four steps are trained iteratively until the model converges.

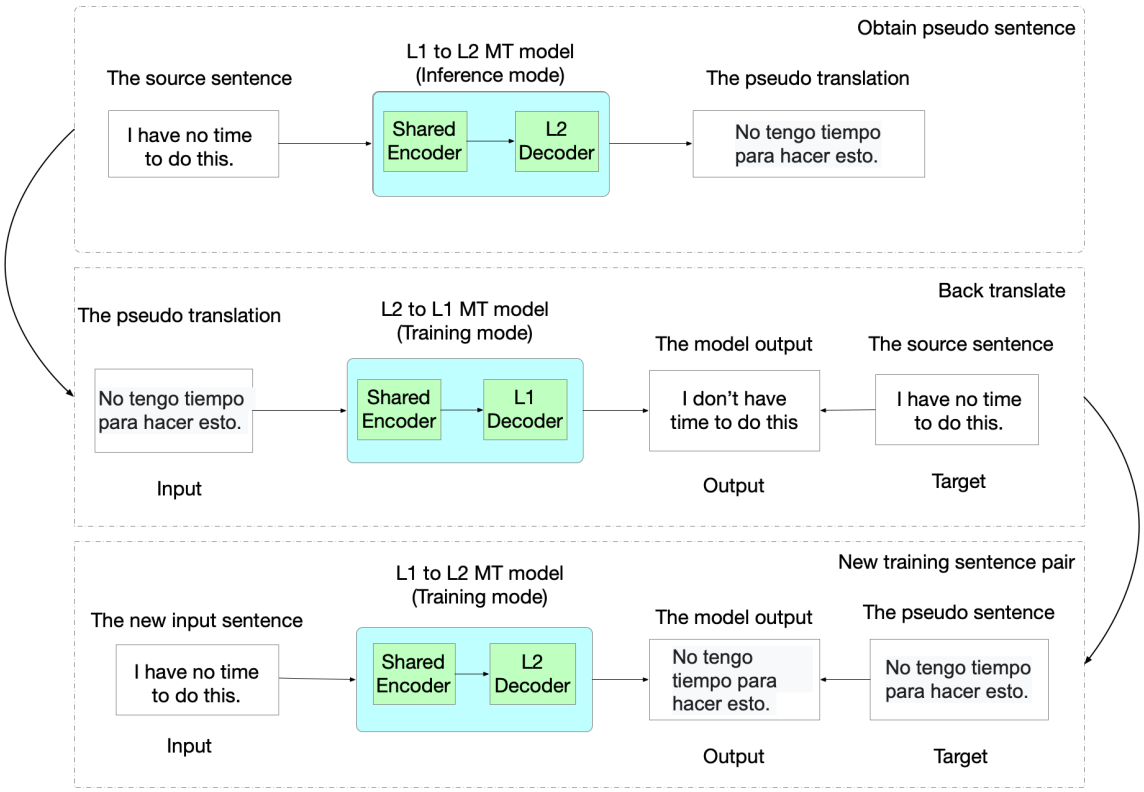



**Figure 9:** Back-translation process.

### 1.3.1 Challenges with UNMT

Performance is a key challenge for UNMT when using cross-lingual word embeddings. The accuracy of the cross-lingual word embeddings has a tendency to be weaker than expected under certain specific circumstances. These circumstances are as follows:

- **Learning cross-lingual word embeddings for languages in different language families.** Currently proposed cross-lingual word embeddings are mostly linear-mapping-based methods. These kinds of methods follow a key assumption, which is that words with similar meanings should share similar geometric arrangements between their monolingual word embeddings. This suggests that there is a linear relationship between languages. However, this assumption does not hold for all language pairs, especially for those word pairs in different language families like English-Chinese or English-Finnish.

- **Learning cross-lingual word embeddings for low-resource languages.** Current word embeddings approaches require large parallel dictionaries as training data to obtain reasonable results. However, those dictionaries are hard to obtain for low-resource languages. This challenge makes it even harder to learn cross-lingual word embeddings in low-resource languages.

Some language pairs have both of those challenges (e.g., English-Finnish), which makes the performance of the cross-lingual word embeddings worse than that of the rich-resource language pairs.

## 1.4 The End-to-End Speech Translation

Speech Translation (ST) is a process of directly translating from spoken language into text of another language. ST research dates back to 1990s, starting with the loosely coupled models, which then moved to end-to-end models in recent years. ST has a wide range of applications like travel assistance (Takezawa et al., 1998), video sub-titling (Saboo and Baumann, 2019) and lesson translation (Fügen, 2009).

### 1.4.1 The Cascaded Model

The cascaded model was the only model in existence for speech translation until last decade. In a cascaded model, an Automatic Speech Recognition (ASR) model and an MT model are separately built and trained. Then the best result of an ASR model is used as the input to the MT model. Earlier cascaded models had deficiencies where the ASR model and the MT model were trained on different modality data. This posed an *error propagation issue* since the output of the ASR would generate high levels of noise for the MT system (Ruiz and Federico, 2014). Therefore, in recent years research focus has turned to end-to-end models.

### 1.4.2 The End-to-End Model

Due to the success of the end-to-end MT models, the end-to-end ST approaches have gained numerous research interests because both the ST and MT models have a similar inner time and sequence structure. Therefore, there are some studies that attempted to directly use the end-to-end MT models to run the ST tasks. Berard et al. (2016) adapt the attention-based MT model to ST tasks and achieve reasonable results; Bérard et al. (2018) extend the end-to-end model to a convolutional encoder in order to better extract the voice features. Wang et al. (2020) adapt the state-of-the-art model of MT, the transformer model, to ST tasks and gain a competitive result compared to cascaded models.

My work considers the end-to-end ST as a sub-field of NMT for three reasons:

1. It can be observed that the timeline of end-to-end ST model is similar with that of NMT.

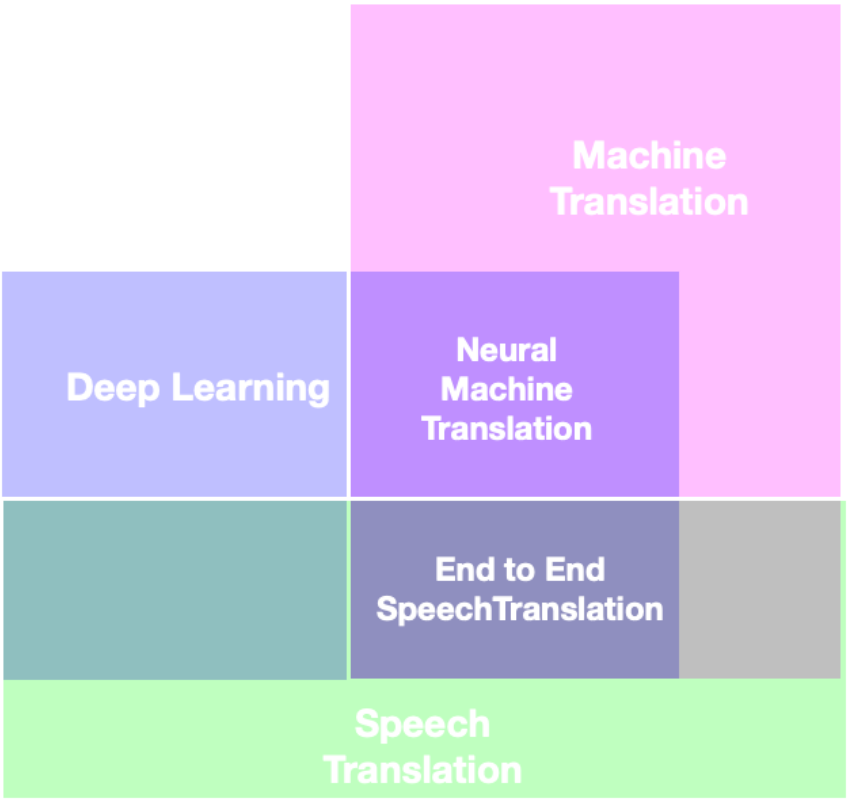



**Figure 10:** The end-to-end speech translation research area.

2. The nature of NMT and end-to-end ST have both similar time sequence data and language translation tasks.

3. The research direction of NMT and end-to-end ST are similar in design structure. Transformer is the popular structure for both end-to-end MT and end-to-end ST models.

I illustrate an overview of my field of research in this study in Figure 10. Chapter. 6 discusses these models in detail.

### 1.4.3 Challenges with End-to-End ST

The main challenge of end-to-end ST is data insufficiency. The speech and its corresponding translated texts are difficult to obtain. The data scarcity issues lead to an immense challenge in translating from speech to text, comparing to text-to-text translation, as performed by the standard MT task. Therefore an important research direction for end-to-end ST is transferring knowledge from the MT task to the ST task (Gaido et al., 2020a; Liu et al., 2019). Additional alternative research direction aiming to alleviate this issue is different data augmentation (Park et al., 2019).

## 1.5 Contributions

I make four key contributions in this dissertation:

1. **I investigated the non-linearity in cross-lingual word embeddings.** The investigation includes developing cross-lingual word embedding for languages within the same language families and different language families. Non-linearity can better describe the relationship between languages than linear approaches. I also provided the English-Chinese dataset with pre-trained word embeddings, a dictionary, and corresponding cross-lingual word embeddings.

2. **I demonstrated how to improve the performance of low-resource cross-lingual word embeddings.** I propose a novel non-linearity-based unsupervised cross-lingual word embedding approach, which learns cross-lingual word embeddings without guidance from dictionaries. Principally, I provide a further non-linear mapping in the current unsupervised cross-lingual word embedding approach. The learned mapping can improve the performance of the cross-lingual word embeddings learned by unsupervised methods, especially the cross-lingual word embeddings of low-resource languages.

3. **I developed the ensemble method in cross-lingual word embeddings**. The investigation consists of two aspects. First, the approach of combining both linear and non-linear methods demonstrates that the proposed ensemble method provides a robust and improved result. Secondly, I further extend the ensemble method to a new word consistency-based strategy, extracting information more effectively through different systems and ultimately achieve better word translation results.

4. **I investigated how to more effectively transfer knowledge from an MT model to a ST model.** I propose a mutual-learning scenario where knowledge transfer takes place in both directions (as opposed to one directional knowledge transfer in traditional knowledge distillation approaches). My work finds that ST can learn knowledge from MT, and vice-versa, utilising the proposed approach, MT can learn knowledge from the ST model effectively. The experiments demonstrate that the proposed approach is more beneficial for translation performance than the currently popular teacher-student models.

## 1.6 Dissertation Structure

The dissertation is composed of seven themed chapters.

- **Chapter 1**: This chapter describes the fundamental theory of machine translation and its related topics.

- **Chapter 2**: This chapter contains the general explanation and definitions within my field, covering deep learning, NLP and word representation technologies.

- **Chapter 3**: This chapter describes my work published in *the 12th Language Resources and Evaluation Conference (LREC 2020)* (Zhao and Gilman, 2020). In this chapter, I give a detailed review of mapping-based cross-lingual word embedding methods. I describe my investigation of non-linearity to cross-lingual word embeddings and provide corresponding experiments.

- **Chapter 4**: I present my proposed approach: non-linearity based unsupervised cross-lingual word embedding. I describe the pipeline of the general unsupervised method and that of my proposed strategy. I give the corresponding analysis and experimental details[2]. The work in this chapter is submitted to *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* and it is under review.

- **Chapter 5**: Here, I provide findings into the feasibility of ensembling linear and non-linear methods. I define a new concept: word consistency, and provide a word consistency-based method together with its corresponding experiments and results. The work in this chapter is preparing to submit to *2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL2021)* and it is under review.

- **Chapter 6**: In this chapter, I present my proposed approach: mutual learning improves end-to-end speech translation. I introduce the pipeline of the cascaded and the end-to-end speech translation. I describe my investigation of the mutual-learning-based speech translation. Also, I give the corresponding analysis and ex-

[2]Covid-19 emerged during my scholarly visit to Spain. The border of New Zealand was closed, so I could only go back to China. The host institute in Spain and Massey University shut down during this period. I had limited access to my data and could not fully incorporate the En-Zh dataset into the experiments in Chapters 4 and 5 as planned. However, this does not impact the corresponding research conclusions.

perimental results. This work is published in *2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)* (Zhao et al., 2021b)

- **Chapter 7**: I conclude this dissertation by summarising all my work and discussing future research directions arising from my study.

CHAPTER 2

# Background and Theoretical Foundations

This chapter describes related work in the areas of machine learning, deep learning, neural networks, language modelling and word embedding methods. It also discusses current dominant technologies in language modelling and word embedding research.

## 2.1 Machine Learning

*Machine learning* can be defined as an algorithm that can learn how to perform a certain task from example data (Goodfellow et al., 2016). Applications driven by machine learning algorithms greatly influence many aspects of our life, from social networks to e-commerce. Its presence grows increasingly in human-machine interaction through smart devices like smartphones and computers.

Machine learning has three key concepts: a task T, an experience E, and a performance measure P. Machine learning aims to improve T's performance P through some experience E (Mitchell et al., 1997).

Generally speaking, machine learning algorithms can be categorised into those that use supervised and unsupervised learning. A critical difference between them is whether labelled data is required.

A supervised learning algorithm is a learning algorithm that requires both input and the corresponding output, so that it can be trained in a *supervised* manner. For instance, an image classification task could aim to classify whether an image is of a dog or a cat. The input is an image and the associated output are two classes: 'dog' or 'cat'. Another example of supervised learning is NMT. For supervised NMT tasks (e.g. English translated to Spanish), the input of an NMT system is a sentence from the source language, e.g.:

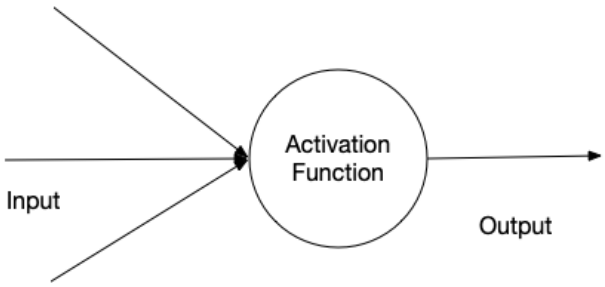



**Figure 11:** A neuron of a neural network.

*I like to play basketball.*

and the associated output is a sentence in target language with the same meaning:

*Me encanta jugar baloncesto.*

An NMT system learns through a corpora, consisting of billions of those sentence pairs. My proposed methods: the non-linear cross-lingual word embedding method, ensemble-based cross-lingual word embedding method and mutual-learning ST method (see Chapter 3, 5 and 6) can be considered as *supervised algorithms*.

Unlike supervised learning algorithms, *unsupervised learning* algorithms attempt to learn knowledge from only inputs without any corresponding outputs. As a general concept, unsupervised learning refers to learning algorithms that do not require manually annotated training data (Goodfellow et al., 2016). Typical unsupervised learning algorithms are principal components analysis (PCA) and k-means clustering. The UNMT and my proposed unsupervised cross-lingual word embeddings (see Chapter 4) can be seen as unsupervised learning algorithms.

## 2.2 Neural Networks

Neural networks, in particular ones labelled as 'deep learning' (LeCun et al., 2015), are an essential component of NMT systems. This section briefly introduces neural networks.

A *neuron* is the basis of a neural network. As shown in Figure 11, the input feature vector $x$ is linearly mapped before an *activation function* $\sigma$ is applied to produce the output vector $y$:

$$y = \sigma(w^\intercal x + b) \tag{2.1}$$

where $w$ are the weights of each input feature and $b$ are their corresponding biases. The activation function is generally non-linear and some popular functions are Sigmoid (see

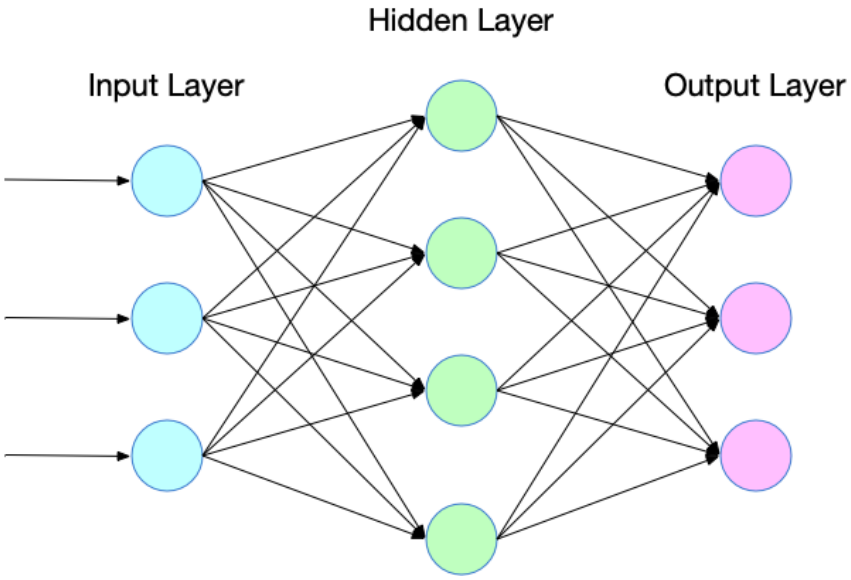



**Figure 12:** The structure of a fully-connected feed-forward neural network.

Equation 2.2) and ReLU (see Equation 2.3) (Ergen and Pilanci, 2021).

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.2}$$

$$\sigma(x) = max(0, x) \tag{2.3}$$

Rectified Linear Unit activation function, also called ReLU, is a function of choice in almost all recent architectures, as it helps to deal with *vanishing gradients* issue (Lu et al., 2019): the gradients gradually become smaller and smaller as they are propagating through multiple network layers during training and hence weight updates tend to zero, making training extremely slow or even stall completely. This issue happens more often when dealing with time sequence data like sentences and voice sequences in MT and ST tasks. This activation function is used more often in current MT and ST tasks.

A basic neural network architecture, the so-called multi-layered perceptron (MLP), consists of three components: the input, hidden, and output layers. The first layer of a neural network is input layer. An output layer is the last layer of a neural network, and other layers between the input and output layers are hidden layers. An MLP may have multiple hidden layers. This network is fully connected, i.e. every neuron in a layer is connected (i.e. takes input from) with every neuron in the previous layer (see Figure 12).

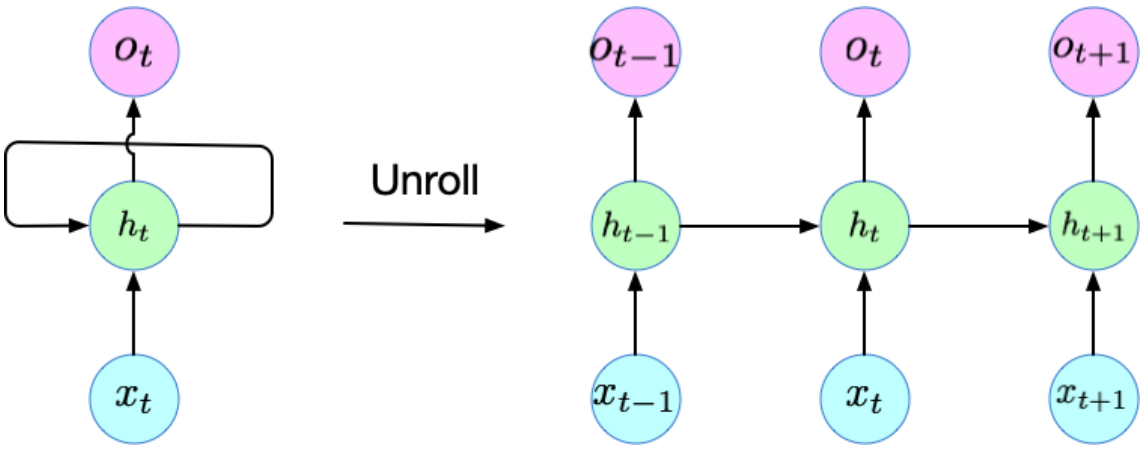



**Figure 13:** The structure of a recurrent neural network.

### 2.2.1 Recurrent Neural Network

Recurrent Neural Networks (RNN) is naturally a better fit for modelling sequence data than feed-forward networks due to their sequential nature. As shown in Figure 13, during each time step, an RNN cell/neuron operates on an element of the input sequence, as well as its own hidden state–this hidden state is the major difference to a fully-connected feed-forward network.

A critical challenge for RNNs is handling long dependencies. When an input sequence is too long (e.g. a long sentence or a long voice feature sequence), the gradients in the back-propagation process may either vanish or explode (Goodfellow et al., 2016). Several variants of RNN were proposed to address this issue. The most successful variants are Long-Short-Term-Memory (LSTM)-based RNN (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU)-based RNN (Chung et al., 2014).

As discussed in the previous chapter, RNNs are a popular choice in modelling MT and ST tasks. However, RNN-based models can not be easily sped-up through parallelisation, due to their sequential computation (time-steps) nature. Thus, the current models for MT and ST tasks are transformer-based models (see Section 1.2.3).

### 2.2.2 Other Variants of Neural Networks

Many other neural networks are used in the deep learning field, like Recursive Neural Networks (Guo et al., 2019), Convolutional Neural Networks (CNN) (LeCun et al., 1995), Spiking Neural Networks (SNN) (Ghosh-Dastidar and Adeli, 2009), *Transformer* (Vaswani et al., 2017), etc.

## 2.3 Language Modelling

Language modelling aims to estimate the probability of a sequence of words in a sentence, and it is fundamental for many NLP tasks. For example, a language model (LM) can learn word representations (Bengio et al., 2003) or serve as initialisation of other downstream tasks like MT, NLI and NLU (Devlin et al., 2019). In fact, neural machine translation can be seen as a direct extension of *neural language modelling* (see Section 2.3.2).

### 2.3.1 The N-gram Model

The N-gram model is a conventional method for learning LM. It performs a sliding window operation of size N over words in a text, forming sequences of word fragments of length N.

Given a sentence with length $m$ from a corpus: $S = (w_1, w_2, ..., w_m)$, where $w_i$ is a word in this sentence, the probability of S can be estimated as $P(S) = P(w_1, w_2, ..., w_m)$. According to Bayes' rule, $P(S)$ can be reformulated as:

$$P(w_1, w_2, ..., w_m) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)...P(w_m|w_1, w_2, ...w_m) \tag{2.4}$$

However, using Equation 2.4 to calculate long sentence probability is complicated and expensive. A common assumption used to simplify this is: a word is related to only $n$ previous words:

$$P(w_1, w_2, ..., w_m) = \prod_{i=1}^{M} P(w_i|w_{i-1}, w_{i-2}, ..w_{i-(n-1)}) \tag{2.5}$$

For instance, when $n = 2$, Equation 2.5 turns into:

$$P(w_1, w_2, ..., w_m) = \prod_{i=1}^{M} P(w_i|w_{i-1}) \tag{2.6}$$

and represents the bi-gram model.

The N-gram model is widely used in statistical methods for natural language processing, for instance, statistical machine translation, speech recognition and text classification (Liu and Yin, 2020; Koehn, 2009; Jelinek, 1997).

### 2.3.2 Neural Language Modelling

Neural Language Modelling (NLM) is effectively a combination of neural networks and LM. Bengio et al. (2003) was the first to propose NLM, which uses a feed-forward neural

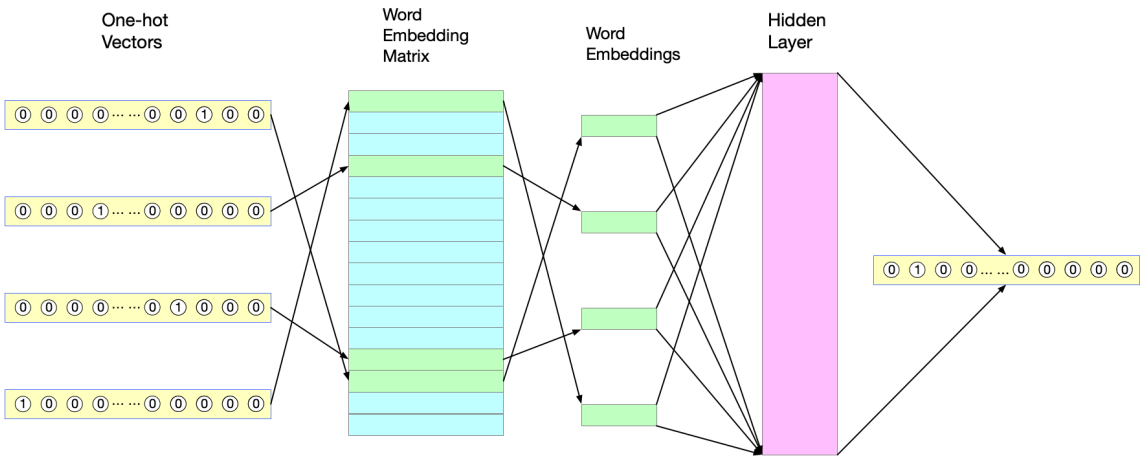



**Figure 14:** Neural language model. The context words with one-hot vector form are first mapped to word embedding vectors through a look-up table, then the word embeddings are used as input to predict the next word.

network to predict the next word using previous $n$ words.

For example, given a context word sequence $(w_{i-4}, w_{i-3}, w_{i-2}, w_{i-1})$, neural language modelling aims to predict the next word $w_i$ of this context word sequence. As illustrated in Figure 14, the words are represented as one-hot vectors. First, those vectors are mapped into a continuous representation called *word embedding* by multiplying them by the *word embedding matrix*. Subsequently, the word embeddings serve as the input to a feed-forward neural network that predicts the next word $w_i$ of the context sequence. During training, the aim is to learn model parameters $\theta$ of $f(w_{i-1}, ..., w_{i-n+1}; \theta)$ by minimising the log-likelihood:

$$L = \frac{1}{m} \sum_{i} \log f(w_i, ..., w_{i-n+1}; \theta) \tag{2.7}$$

where $f$ is a feed-forward neural network that maps the previous $n$ context words in a word sequence to the current word $w_i$ and $m$ is the length of the word sequence. Subsequently, word embedding matrix can be seen as the part of the parameter set and also optimised as a part of $\theta$.

An exciting finding of this work is a new way to represent words. The result of multiplying a one-hot vector and the word embedding matrix can be seen as a word representation, often called word embeddings (words embedded in a vector space). These new word representations, learnt by the NLM task are more effective than simple one-hot encoding and have since been widely adopted in many NLP applications.

An alternative neural network architecture frequently used for NLM is the recurrent neural network. The advantage of an RNN over a feed-forward neural network is that the

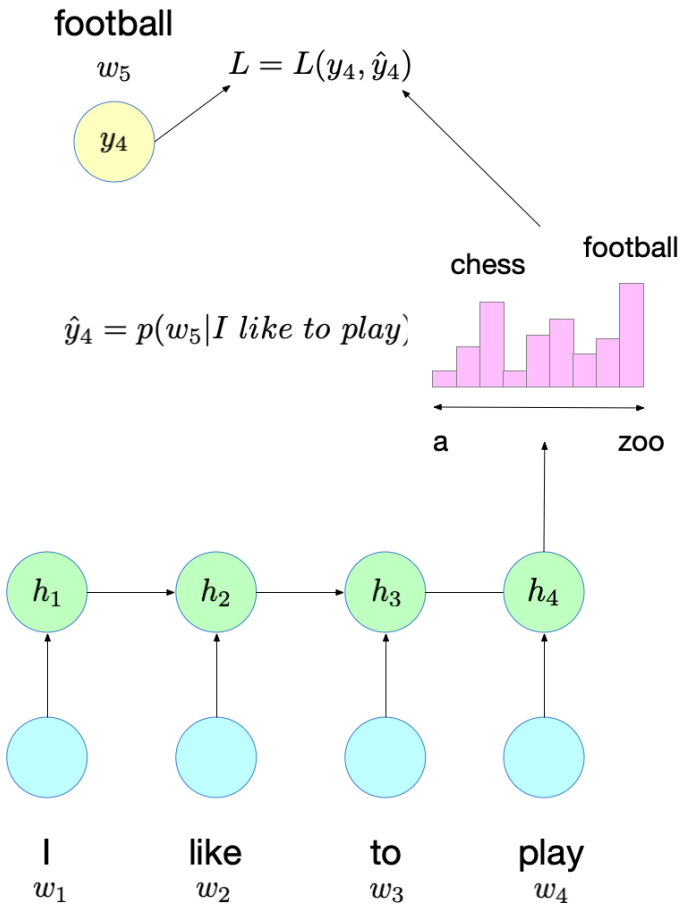



**Figure 15:** Training process of the RNN neural language model at time step four with an input sentence 'I like to play football'. The input of the RNN is one-hot vector representation of words in the input sequence. The target $y$ is also a one hot representation. In each time step, RNN calculates the probability distribution of the next word.

RNN can model text sequences of any length. As illustrated in Figure 15, in each time step of RNN, the input word $w_{n-1}$ is used to predict its following word $w_n$.

Interestingly, when predicting the next word of a sentence using an RNN, the hidden layers of the RNN are able to 'remember' information from all the previous steps. The last hidden layer of RNN can be considered to be a compressed representation of the sentence up to this point and used in downstream tasks, such as neural machine translation (that was introduced in Chapter 1).

## 2.4 Word Representation

Word representation is a foundation of most NLP related tasks like MT, natural language inference (NLI) and natural language understanding (NLU) (Devlin et al., 2019).

### 2.4.1 One-hot Vector Representation

One-hot word representation often serves as the original input of an NLP model, turning words into digital numbers in the first place. Specifically, a word is represented by a dictionary-length vector. In this vector, the word's corresponding index in the dictionary is 1, and the remaining elements are 0. For example, given a corpora consisting of two documents:

(a) *Luke likes to play tennis. Mike likes to play tennis, too.*

(b) *Mikel likes to watch movies.*

First a list of unique words is composed (the vocabulary). Each word can now be represented by its index in the vocabulary list, e.g. { 'Luke':1, 'likes':2, 'to':3, 'play':4, 'tennis':5, 'Mike':6, 'also':7, 'too':8, 'Mikel':9, 'watch':10, 'movies':11 }. One-hot vector encoding of these indices now serves as the word representation e.g. the word *'Luke'* is represented by:

[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Thus, document (b) can be represented by a matrix (where rows are word embeddings that represent the words):

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.8}$$

It can be observed that when the vocabulary of a corpus grows larger, the computational complexity of working with these word representations grows exponentially, making it difficult to use one-hot vector representation for large corpora.

### 2.4.2 Distributed Word Representation

Word embedding approaches started with an unexpected but exciting intermediate representation that came out of a language modelling task proposed by Bengio et al. (2003). This intermediate representation of a neural network helps represent words (Bengio et al., 2003).

Inspired by Bengio et al. (2003), many neural network-based models were proposed to learn word embeddings. The most successful work was conducted by Mikolov et al. (2013b), which is called *word2vec*. Word2vec serves as one of the fundamental technologies used in the work described in Chapters 3, 4 and 5.

Compared to traditional one-hot vector representation, the distributed representation method has advantages:

1. Word embeddings are dense representations. Typical word representation methods, e.g. one-hot vector representation, represent words in a much higher dimensional space, making them impractical when dealing with large corpora.

2. Words with similar meanings are expected to have similar word embeddings - the relationship among embedding pairs can be leveraged to qualify syntactic and semantic relationships. This unique property allows us to measure word analogy effectively.

Word2vec consists of two alternative models to learn the word embeddings, which are called *skip-gram* model and *continuous bag-of-words* model (or *CBOW* in short).

**Skip-gram model** aims to learn a word's representation by predicting its surrounding words. As illustrated in Figure 16, one-hot representation of a word $w_i$ is used as an input and the objective is to predict its surrounding words $(w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2})$. After training the skip-gram model, the mapping matrix $M \in R^{V \times N}$ between the input layer and the hidden layer will serve as a *word embedding matrix*, where the $i_{th}$ entry of the $M$ represents the $i_{th}$ word.

**CBOW model** aims to learn a word embedding by **predicting a word from its surrounding words**. Therefore in contrast with skip-gram model, the input is a specific word $w_i$'s surrounding words $(w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2})$ and the $w_i$ is the target. Figure 16 illustrates the CBOW model.

Both Skip-gram model and CBOW model are called *word2vec*. Word2vec is better than the previous methods because:

- It belongs to the unsupervised learning paradigm. To learn word embeddings using this method, no labelled data is required - just a monolingual corpora.

- Word analogies can be evaluated by vector arithmetic.

Mikolov et al. (2013b) give a famous example to exhibit word2vec's property: Subtracting the embedding that represents word 'man' from 'king' and adding an embedding that represents the word 'woman' results in a vector whose nearest neighbour is an embedding that represents the word 'queen':

$$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} \approx \overrightarrow{\text{queen}}$$

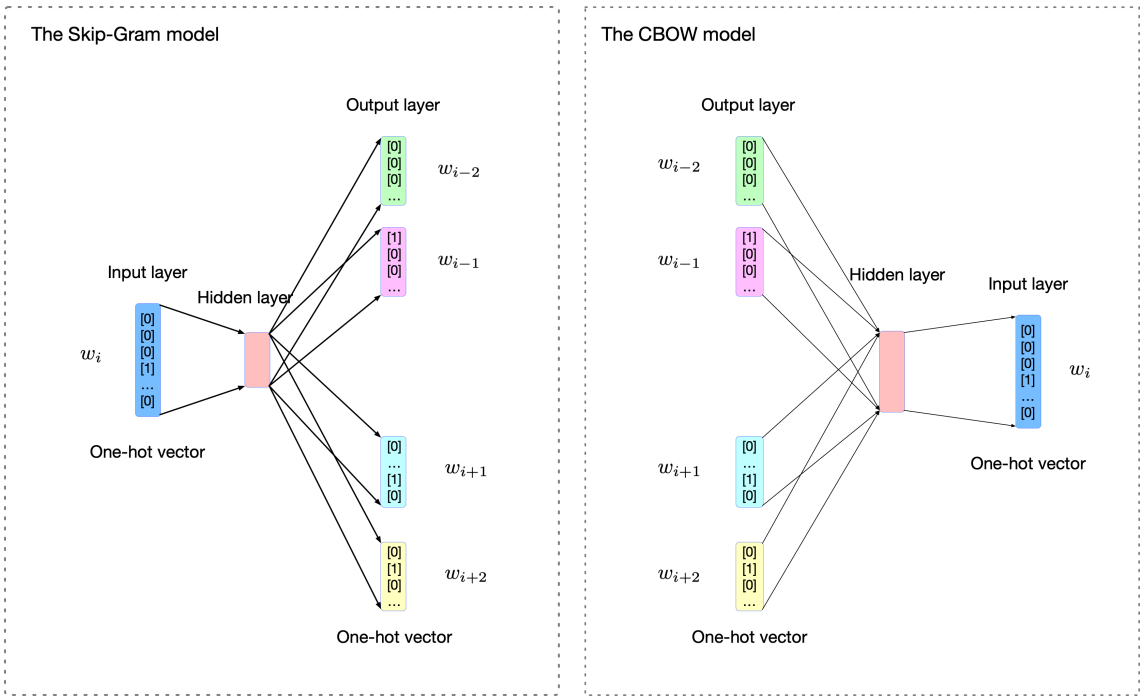



**Figure 16:** Skip-gram model and CBOW model.

which suggests that word meaning is learnt from its context and can 'be compared' mathematically. It has been shown that if those learnt word embeddings are projected into a two-dimensional space using some dimensionality reduction methods (e.g. PCA), **words with similar meanings share similar geometric arrangements** (Mikolov et al., 2013a). This finding is the underpinning concept of cross-lingual word embedding methods, and I will further discuss it in the next chapter. The success of word2vec has driven the distributed representation of words to become the most effective method to represent words.

**Global Vectors for Word Representation (GloVe)** is an important improvement on word2vec (Pennington et al., 2014). One of the word2vec model's weaknesses is its failure to extract global information from the context (Liu et al., 2020). GloVe captures word features directly from the overall corpus statistics, which can better extract global information.

**Embeddings from Language Models (ELMo)** is another popular word representation method (Peters et al., 2018). It addresses two key challenges in representing words:

1. It is difficult to capture high quality word representation that contains syntax and sentiment information

2. Adapting the word representation to various tasks in NLP is difficult.

ELMo model addresses these issues by learning knowledge from language modelling tasks.

### 2.4.3 Transformer-based Models

Thanks to the recent success of transformers, some new word embedding methods have been proposed. The most successful model is proposed by Devlin et al. (2019) and is called **Bidirectional Encoder Representations from Transformers (BERT)**. It is also a representation learning method through language modelling. Through pre-training using a modified language modelling task, the word embedding matrix learnt by the transformer can more effectively represent the words and obtain better performance in downstream tasks.

Inspired by pre-trained transformer models, many similar works are proposed like GPT2 (Radford et al., 2019). Nowadays, the word representation approaches are turning towards higher integration with downstream tasks. However, pre-train models are criticised by their expensive cost and complicated structure (Dickson, 2020). The future of learning word representations can benefit from the light and cheap models.

CHAPTER 3

# Non-Linearity in Cross-Lingual Word Embeddings

## 3.1 Introduction

Word2vec has gained numerous successes in many NLP tasks. The word embeddings learnt from a monolingual corpus are also called *monolingual word embeddings*. Monolingual word embeddings widely serve as features in many NLP tasks such as machine translation and text classification (Bengio et al., 2003).

Mikolov et al. (2013a) made an important observation: monolingual word embeddings constructed from different languages have similar relative geometric arrangements for the embeddings that represent similar concepts in the different languages (Mikolov et al., 2013a). This finding makes it possible to find a transformation to map word embeddings in different feature spaces (constructed from different languages independently) into a shared feature space. The projected vectors are called *cross-lingual word embeddings*. Compared to monolingual word embeddings, cross-lingual word embeddings poses a better representation ability when applied in cross-lingual tasks, due to two advantages:

1. Cross-lingual word embeddings enable us to compare the contextual usage of words from different languages. This is the key to NLP tasks like machine translation and bilingual lexicon induction.

2. Cross-lingual word embeddings allow one to transfer knowledge between languages. For instance, through the use of cross-lingual word embeddings one is able to transfer knowledge from a high-resource language to a low-resource language. This ability is important to knowledge transfer tasks like unsupervised neural machine translation (Artetxe et al., 2017).

Recent works on cross-lingual word embeddings have been mainly focused on linear-mapping-based approaches, where pre-trained monolingual word embeddings are mapped into a shared vector space using a linear transformation. The linear-mapping-based approaches follow a key assumption, first stated by Mikolov et al. (2013a): **The monolingual word embeddings in different languages are expected to share similar geometric arrangements, in terms of the relative location of words that represent similar concepts (in different languages)**. This assumption suggests that there is a linear relationship between those word embeddings in different languages. However, this assumption may not hold for all language pairs across all semantic concepts.

This chapter investigates whether the non-linear mapping can better describe the relationship between different languages by utilising Kernel Canonical Correlation Analysis (KCCA) and Deep Canonical Correlation Analysis (DCCA). I formally define cross-lingual word embeddings and describe the pipeline used to construct them; then, I discuss my proposed approach. I explain my experiments and experimental results, which show an improvement over the current state-of-the-art in both supervised and self-learning scenarios, confirming that the use of a non-linear mapping can be a better alternative to describe the relationship between languages.

## 3.2 Description of Notation

This section introduces several key concepts in cross-lingual word embeddings. But first, I outline the notation used to describe the concepts. Table 3.1 summarises the notation used throughout this chapter.

$V_s$ and $V_t$ represent the vocabularies of the source and target languages, respectively, with $\tilde{N}_s$ and $\tilde{N}_t$ words each. A matrix whose rows are word embeddings is referred to as the **word embedding matrix**. The word embedding matrix of the source vocabulary is denoted by $\tilde{X}_s \in \mathbb{R}^{\tilde{N}_s \times d_s}$, in which each row is a word embedding representing one word from the source vocabulary. Similarly, the word embedding matrix of the target language is denoted by $\tilde{X}_t \subset \mathbb{R}^{\tilde{N}_t \times d_t}$. The dimension of the source and target word embeddings are denoted by $d_s$ and $d_t$.

A bilingual dictionary $D$ is two ordered lists of words, where for each word in the first list, a word with the same index in the second list is its target language translation. A sample dictionary is shown in Figure 17. From this dictionary, the source and target word translations can obtained. Then the two word embedding matrices $X_s \subset \tilde{X}_s$ and $X_t \subset \tilde{X}_t$

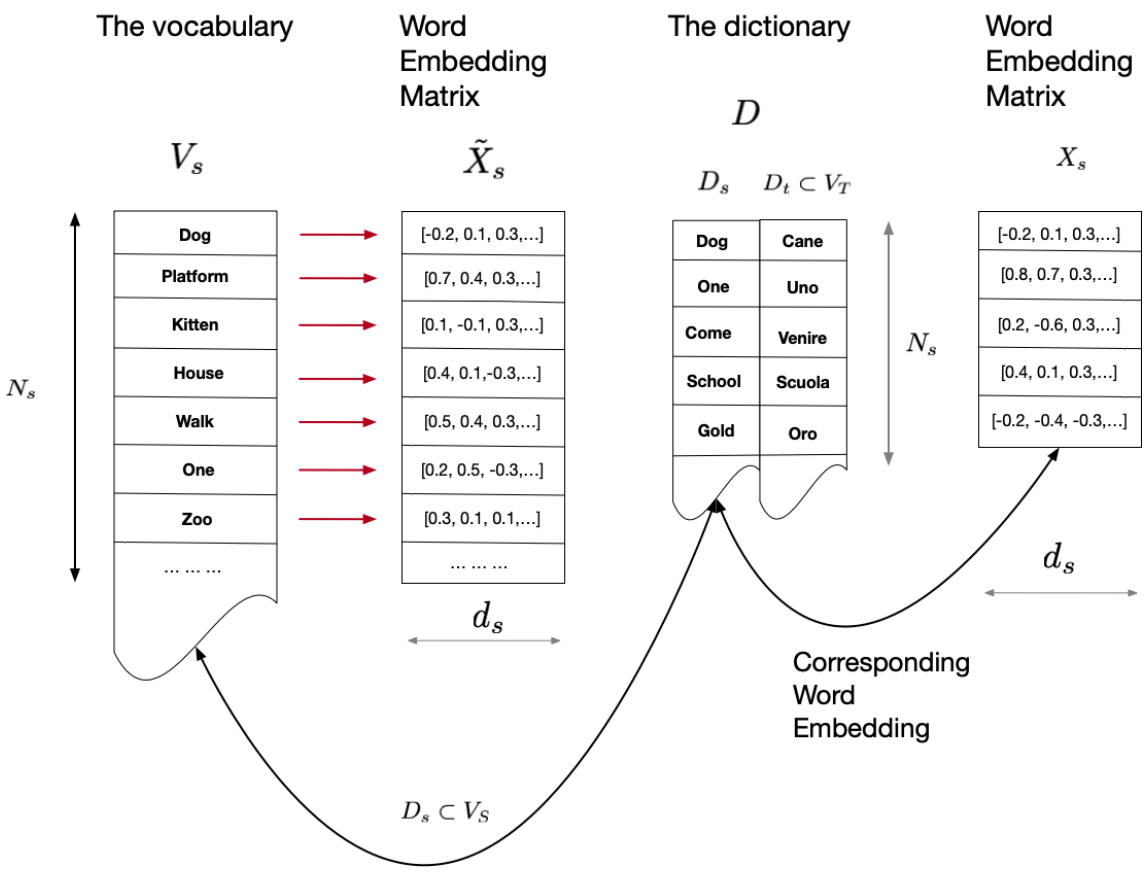



**Figure 17:** A visual representation of the concepts used in this chapter. $V_s$ is the vocabulary of the source corpus. The $\tilde{X}_s$ is the word embedding matrix of the vocabulary. Each row of word embedding matrix $\tilde{X}_s$ is a word vector representing a word in the vocabulary. Given a bilingual dictionary D, the source words of the dictionary are used to formulate a word embedding matrix $X_s \in \tilde{X}_s$. Each row of $X_s$ is a word embedding representing a source language word from the dictionary.

can be formed. Each row of $X_s$ or $X_t$ represents its corresponding word in the given dictionary. The relationships between $\tilde{X}_s$, $\tilde{X}_t$, $X_s$ and $X_t$ are illustrated in Figure 17.

## 3.3 Related Work: Mapping-based Approaches

One type of cross-lingual word embedding approaches, popular because of its simplicity, is the mapping-based approach. It maps pre-trained monolingual word embeddings into a shared vector space using a transformation (Artetxe et al., 2018a, 2016; Faruqui and Dyer, 2014; Mikolov et al., 2013a). Existing methods are based on a key assumption: embeddings of words with similar concept in different languages share similar geometric arrangements (Mikolov et al., 2013a), which suggest that there is a linear relationship between the word embeddings representing similar concepts in different languages. Figure 18 illustrates this approach, where the dimensions of the word embeddings have been reduced to two for visualisation purposes. It can be observed that the word embeddings

**Table 3.1:** Used notations of the cross-lingual word embeddings.

| Symbol | Meaning |
|---|---|
| $s$ | Source language |
| $t$ | Target language |
| $V_s$ | Vocabulary of the source language |
| $V_t$ | Vocabulary of the target language |
| $D$ | Bilingual dictionary of the source and target language |
| $x$ | Word embedding |
| $d_s$ | Dimensionality of source language word embeddings |
| $d_t$ | Dimensionality of target language word embeddings |
| $\tilde{N}_s$ | Source language vocabulary size |
| $\tilde{N}_t$ | Target language vocabulary size |
| $N_s$ | Source language dictionary size |
| $N_t$ | Target language dictionary size |
| $\tilde{X}_s$ | Source language vocabulary word embedding matrix |
| $\tilde{X}_t$ | Target language vocabulary word embedding matrix |
| $X_s$ | Source language dictionary word embedding matrix |
| $X_t$ | Target language dictionary word embedding matrix |
| $W$ | Transformation matrix |
| $w_s$ | Projection vector of a source word embedding |
| $w_t$ | Projection vector of a target word embedding |

of the concepts 'dog', 'cat', and 'cow' in English and Spanish have similar geometric arrangements. Based on this finding, Mikolov et al. (2013a) conclude that there exists a linear mapping that can project word embeddings from different languages into a shared vector space.

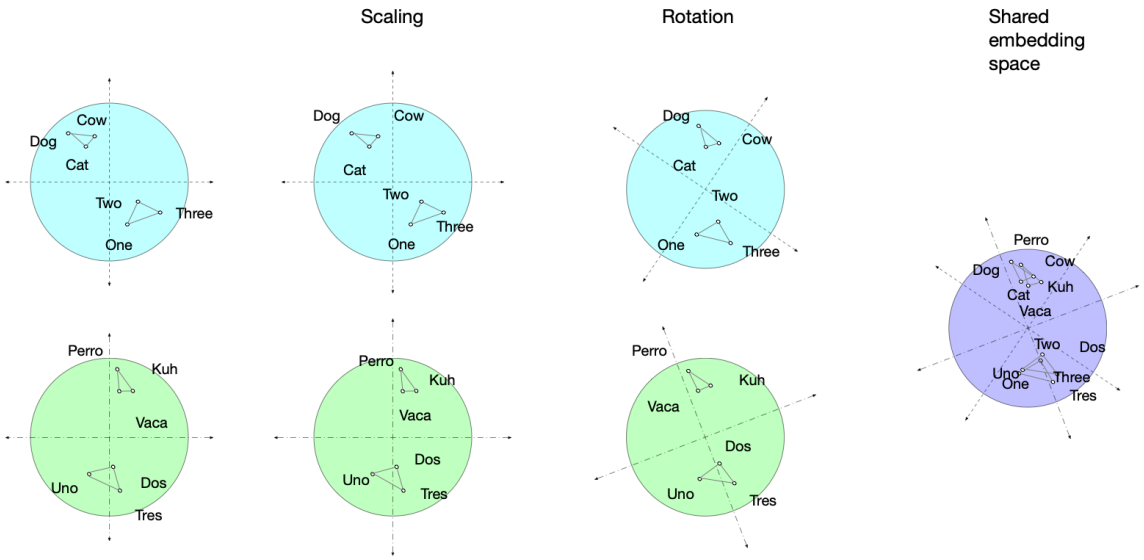



**Figure 18:** Visual demonstration of words with similar meaning having similar geometric arrangements (embeddings compressed to two dimensions for visualisation purposes). Linear transformations, like rotation and stretching, can map word embeddings (from different feature spaces) into a shared word embedding space.

### 3.3.1 The Original Linear Mapping-based Method

The earliest and most influential approach for learning the linear mapping is proposed by Mikolov et al. (2013a). In their work, a linear mapping is learnt to project a word embedding $\mathbf{x}$ from the source language feature space to that of the target language. The mapping $W$ is learnt by minimising the mean square error (MSE) between the mapped word embeddings matrix $WX_s$ and the target language word embedding matrix $X_t$:

$$W = \underset{W}{\operatorname{argmin}} \|X_s W - X_t\|_2^2 \tag{3.1}$$

where $X_s$ and $X_t$ are the word embedding matrices of dictionary words in source and target languages. The authors solve this minimisation problem using stochastic gradient descent (SGD).

### 3.3.2 Orthogonal Methods

Xing et al. (2015) further improved the linear-mapping-based approach by constraining the transformation matrix $W$ to be orthogonal:

$$W^\intercal W = \mathrm{I} \tag{3.2}$$

where $(\cdot)^\intercal$ denotes the transpose operation. Under this constraint, an analytical solution to Equation 3.1 can be derived:

$$W = \underset{W}{\operatorname{argmin}} \|X_s W - X_t\|_2^2 \tag{3.3}$$

$$= \underset{W}{\operatorname{argmin}} (\|X_s W\|_2^2 - \|X_t\|_2^2 - 2X_s W X_t^\intercal) \tag{3.4}$$

Because $\|X_t\|_2^2$ and $\|X_s W\|_2^2$ are independent of the value of $W$, Equation 3.3 is equivalent to:

$$W = \underset{W}{\operatorname{argmax}} X_s W X_t^\intercal \tag{3.5}$$

Because $W$ is an orthogonal matrix, based on the optimization rules of matrices, Equation 3.5 is equivalent to:

$$W = \underset{W}{\operatorname{argmax}} \operatorname{Tr}(X_s W X_t^\intercal) \tag{3.6}$$

where $\operatorname{Tr}(\cdot)$ denotes the trace operation. Based on the property of the matrix trace, Equation 3.6 is equivalent to:

$$W = \underset{W}{\operatorname{argmax}} \operatorname{Tr}(X_t^\intercal X_s W) \tag{3.7}$$

Then singular value decomposition (SVD) is applied to $X_t^\intercal X_s$:

$$X_t^\intercal X_s = U S V^\intercal \tag{3.8}$$

Therefore,

$$\begin{aligned} \underset{W}{\operatorname{argmax}} \operatorname{Tr}(X_t^\intercal X_s W) &= \underset{W}{\operatorname{argmax}} \operatorname{Tr}(U S V^\intercal W) && (3.9) \\ &= \underset{W}{\operatorname{argmax}} \operatorname{Tr}(S V^\intercal W U) && (3.10) \end{aligned}$$

Since $V^\intercal W U$ is an orthogonal matrix, this optimisation problem can be seen as maximising the trace of an orthogonal transformation of $S$. An orthogonal transformation of $S$ will be maximised when $V^\intercal W U = I$. Thus, Equation 3.3 can be solved by:

$$W = V U^\intercal \tag{3.11}$$

Then the mapping of $X_s$ is defined as $W_s = U$ and the mapping of $X_t$ is defined as $W_t = V$. The projected word embedding matrices $W_s X_s$ and $W_t X_t$ are then in a shared vector space. In the dissertation, I will refer to this kind of methods as the *SVD-based* methods for simplicity.

### 3.3.3 The CCA-based Approach

Another linear-mapping-based approach utilises Canonical Correlation Analysis (CCA). First, I briefly introduce CCA. Given two multivariate random variables $x_1 \in \mathbb{R}^{d_1}$, $x_2 \in \mathbb{R}^{d_2}$ (i.e. two word embedding vectors), CCA aims to find basis vectors $w_1$ and $w_2$, such that the correlation $\rho$, between the projections onto those basis vectors, $w_1^\intercal x_1$ and $w_2^\intercal x_2$, are mutually maximised:

$$\begin{aligned} w_1, w_2 &= \underset{w_1, w_2}{\operatorname{argmax}} \rho(w_1^\intercal x_1, w_2^\intercal x_2) && (3.12) \\ &= \underset{w_1, w_2}{\operatorname{argmax}} \frac{w_1 C_{12} w_2^\intercal}{\sqrt{w_1 C_{11} w_1^\intercal} \sqrt{w_2 C_{22} w_2^\intercal}} && (3.13) \end{aligned}$$

where $C_{11}$ and $C_{22}$ denotes the covariance of $x_1$ and $x_2$, and $C_{12}$ denotes the cross-covariance of $x_1$ and $x_2$. Since scaling $w_1$ and $w_2$ has no effect on the result of this maximisation problem, its solution is equivalent to maximising just the the numerator of Equation 3.13,

subject to an additional constraint $\sqrt{w_1 C_{11} w_1^T} = \sqrt{w_2 C_{22} w_2^T} = 1$. This constrained optimisation can be solved through the use of Lagrange multiplier method (Lagrangian relaxation). The $w_1$ and $w_2$ are also called the *first canonical components*. This maximisation procedure can be repeated $d$ times to find another set of basis functions $w_1^i$ and $w_2^i$, orthogonal to the previous $i-1$ basis vector pairs. Then two sets of basis vectors are obtained. These two sets of new bases form vector spaces where $x_1$ and $x_2$ are maximally correlated.

There are several solutions to CCA. This dissertation follows the solution provided by Kent et al. (1979). Consider all pairs of basis vectors $w_1^i$ and $w_2^i$, the top $k$ ($k \leq \min(d_1, d_2)$) basis vectors are assembled into columns as matrices $A \in \mathbb{R}^{d_1 \times k}$ and $B \in \mathbb{R}^{d_2 \times k}$. The overall objective of CCA is to obtain:

$$\underset{A,B}{\operatorname{argmax}} \operatorname{Tr}(A^T C_{12} B) \tag{3.14}$$

subject to:

$$A^T C_{11} A = B^T C_{22} B = I \tag{3.15}$$

Where $I$ is a $k \times k$ identity matrix. Let us define a matrix $T$, such that:

$$T = C_{11}^{-\frac{1}{2}} C_{12} C_{22}^{-\frac{1}{2}} \tag{3.16}$$

And apply singular value decomposition to $T$:

$$T = USV^T \tag{3.17}$$

Let $U_k$ and $V_k$ be the first $k$ basis vectors in $U$ and $V$. The optimum is then transferred to the sum of the top $k$ singular values of $T$, and this optimum is attained when:

$$A = C_{11}^{-\frac{1}{2}} U_k \tag{3.18}$$

$$B = C_{22}^{-\frac{1}{2}} V_k \tag{3.19}$$

This solution assumes that the covariance matrices are non-singular. Which in practice can be overcome with regularisation. Given a centred data matrix $\overline{H}_1$, the covariance matrix can be estimated as:

$$\hat{C}_{11} = \frac{1}{k-1} \overline{H}_{11} \, \overline{H}_{11}^T + r_1 I \tag{3.20}$$

In Equation 3.20, $r_1 > 0$ is a regularisation parameter. The regularisation process aims to guarantee the matrix is non-singular, and alleviate the overfitting problem (De Bie and De Moor, 2003).

Faruqui and Dyer (2014) were the first to propose the *CCA-based cross-lingual word embeddings*. They adopt CCA to learn cross-lingual word embeddings from two sets of pre-trained monolingual word embeddings. This work demonstrates that cross-lingual embeddings can improve the performance of several tasks compared to monolingual word embeddings. The method applies CCA to find $W_s$ and $W_t$ projection matrices:

$$W_s, W_t = CCA\ (X_s, X_t) \tag{3.21}$$

where $W_s \in \mathbb{R}^{d_s \times d}$ and $W_t \in \mathbb{R}^{d_t \times d}$ contains the mapping vectors and the dimension of the projected vectors. The learnt mapping can then be used to project the whole vocabulary embeddings $\tilde{X}_s$ and $\tilde{X}_t$ into the new shared embedding vector space:

$$\tilde{X}_s^* = W_s \tilde{X}_s \tag{3.22}$$

$$\tilde{X}_t^* = W_t \tilde{X}_t \tag{3.23}$$

Ammar et al. (2016) extend this work to a multi-lingual scenario, which can map word embeddings from more than two languages into a shared embedding space.

## 3.4 Description of the Mapping-based Cross-Lingual Word Embeddings Pipeline

The pipeline for creation of mapping-based cross-lingual word embeddings can be decomposed into four separate steps: pre-processing, mapping, re-weighting and dimensionality reduction. It has become fairly standard in the community and is described here for completeness.

### 3.4.1 Pre-processing

Each column of the monolingual word embedding matrix $X$ is mean-centred by subtracting column mean from each element of that column. Then each row of the resulting matrix is length normalised, by dividing each element of that row by the row's squared Euclidean norm.

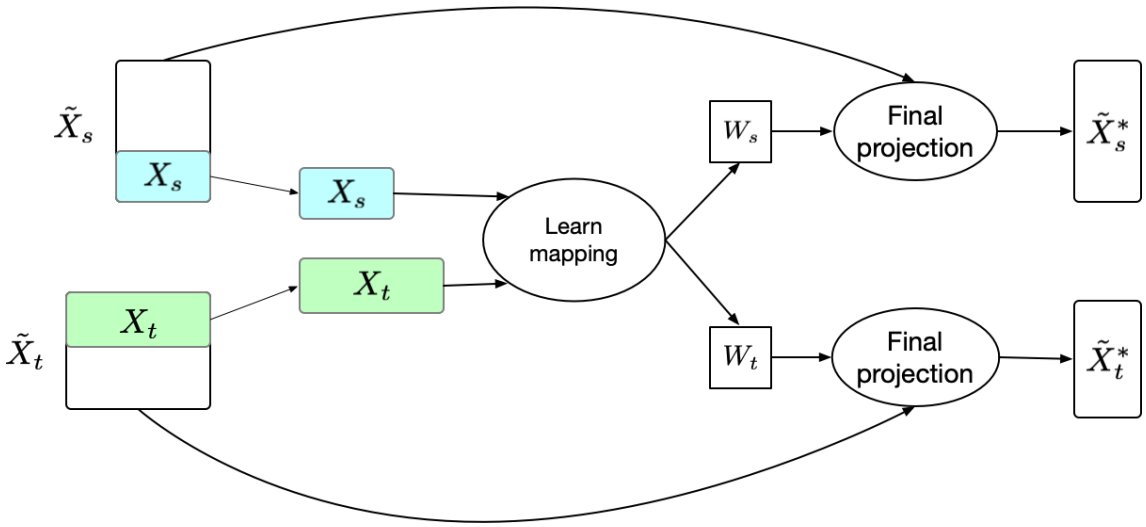



**Figure 19:** Visualisation of the cross-lingual word embeddings construction process using mapping-based approaches.

As discussed before, this step ensures the projections computed in the following mapping process to be orthogonal. Artetxe et al. (2018a) also use additional whitening and de-whitening process, aimed at making each dimension of word embeddings have a unit variance.

### 3.4.2 Mapping

The mapping approaches described in Section 3.3 are applied to pre-processed word embedding matrices in this step. The monolingual word embeddings from different languages are mapped into a shared embedding space. Either Equation 3.18 is used, for the CCA-based mapping method, or Equation 3.11 for the orthogonal mapping-based method.

### 3.4.3 Re-weighting

An experiment conducted by Artetxe et al. (2018a) shows that re-weighting yields better results than a dimensionality reduction process. Re-weighting can smoothly re-scale components of the produced embedding matrix. SVD-based methods re-weight the cross-lingual word embeddings by:

$$\tilde{X}_s^* = \tilde{X}_s^* S \tag{3.24}$$

$$\tilde{X}_t^* = \tilde{X}_t^* S \tag{3.25}$$

where S refers to the singular values from the mapping step.

The re-weighting process was initially proposed by Artetxe et al. (2018a) for use with the SVD-based mapping methods. After mapping, the components of the mapped embedding vectors are re-weighted based on their singular values, which produces stronger cross-lingual embeddings. However, they failed to make re-weighing work with CCA-based mapping. In my opinion, this was one of the reason for them to not include CCA in their work. I successfully adopt the process and apply it to CCA and KCCA. The components of the new embeddings are re-weighted based on their canonical correlations:

$$\tilde{X}_s^* = \tilde{X}_s^* \rho^\zeta \tag{3.26}$$

$$\tilde{X}_t^* = \tilde{X}_t^* \rho^\zeta \tag{3.27}$$

In Equation 3.26 and 3.27, $\zeta$ is a parameter with a default value of 1. Additionally, different language pairs may require different values of this parameter to get optimal results and ideally it shall be tuned.

### 3.4.4 Dimensionality Reduction

In this step, only the first $k$ dimensions of the produced cross-lingual word embeddings are preserved, with dimensions ranked either by decreasing correlation value $\rho$ for CCA-based methods or decreasing value of singular values for SVD-based methods. However, Artetxe et al. (2018a) claim that the re-weighting and dimensionality reduction have an overlapping effect. The dimensionality reduction process can be seen as a special case of re-weighting, where the first $i$ components are weighted by unity and the remaining components are weighting by zero. Previous work has shown that re-weigthing outperforms dimensionality reduction, so the dimensionality reduction process can be discarded in the pipeline (Artetxe et al., 2018a).

## 3.5 Proposed Investigation: Non-Linear Methods in Cross-Lingual Word Embeddings

Two novel methods of introducing non-linearity into mapping-based cross-lingual word embedding approaches are described in this section: Kernel Canonical Correlation Analysis (KCCA) and Deep Canonical Correlation Analysis (DCCA). My investigation into the use of non-linear methods had two objectives:

1. Incorporate non-linearity into mapping-based cross-lingual word embeddings.

2. Investigate the performance of a deep learning method (DCCA) and a machine learning method (KCCA) for this purpose under constrained amount of labelled training data.

### 3.5.1 KCCA

The main limitation of CCA is its linearity. In contrast to CCA, KCCA enables non-linear mapping, by projecting the data into a higher-dimensional space, using a mapping function $\phi$ (see Equation 3.28), before performing CCA in that higher-dimensional space.

$$\phi : p = (p_1, ..., p_d) \rightarrow \phi(p) = (\phi_1(p), ..., \phi_D(p)) \tag{3.28}$$

Equation 3.28 shows a random variable $p$ from the original feature space $\mathbb{R}^d$ mapped by function $\phi$ to a new feature space $\mathbb{R}^D$, where $D \gg d$. However, this new space may be highly multi-dimensional (or even infinitely dimensional) and operating on it directly is computationally expensive (or infeasible). Kernel methods offer a dual representation, which allows to operate on data in a high-dimensional space while avoiding explicitly mapping it into that space using $\phi$. Kernel methods define a pair-wise similarity function, called the kernel function, $K$, such that $\forall p_i, p_j \in p$:

$$k(p_i, p_j) = \langle \phi(p_i), \phi(p_j) \rangle \tag{3.29}$$

where $\langle \cdot, \cdot \rangle$ is the inner product operation. Any machine learning algorithm that can be expressed via such a similarity function of features, rather than the features themselves, can utilise so called kernel trick and work with the new higher-dimensional features (actually their inner products) without ever computing them directly.

Kernelised CCA can be expressed in the following way. Let $a, b \in \mathbb{R}^d$ be two multivariate random variables (e.g. word embeddings) and $\phi(a), \phi(b) \in \mathbb{R}^D$ be the projections of these into the new high-dimensional vector space. Consider data matrices $A \in \mathbb{R}^{N \times D}$ and $B \in \mathbb{R}^{N \times D}$, whose rows contain the sample vectors in the new high-dimensional feature space (i.e. instances of $\phi(a)$ and $\phi(b)$). Equation 3.12 can be rewritten by expressing the co-variance matrices in terms of these data matrices ($C_{aa} = A^\intercal A, C_{bb} = B^\intercal B, C_{ab} = A^\intercal B$):

$$\underset{w_a, w_b}{\operatorname{argmax}} \frac{w_a^\intercal A^\intercal B w_b}{\sqrt{w_a^\intercal A^\intercal A w_a}\sqrt{w_b^\intercal B^\intercal B w_b}} \tag{3.30}$$

If the basis $w_a$ and $w_b$ are expressed as linear combinations of data points using coefficients $\alpha \in \mathbb{R}^N$ and $\beta \in \mathbb{R}^N$:

$$w_a = A^\intercal \alpha \tag{3.31}$$

$$w_b = B^\intercal \beta \tag{3.32}$$

Then the dual representation of the problem can be formulated by substituting Equations 3.31 and 3.32 into Equation 3.30:

$$\underset{\alpha,\beta}{\operatorname{argmax}} \frac{\alpha^\intercal A A^\intercal B B^\intercal \beta}{\sqrt{\alpha^\intercal A A^\intercal A A^\intercal \alpha}\sqrt{\beta^\intercal B B^\intercal B B^\intercal \beta}} \tag{3.33}$$

To solve Equation 3.33, it is not necessary to compute the data matrices in the high-dimensional space, $A$ and $B$, directly, only their inner products $AA^\intercal$ and $BB^\intercal$, which in turn can be expressed via the kernel matrices (Gram matrices) $K_a = AA^\intercal$ and $K_b = BB^\intercal$:

$$\underset{\alpha,\beta}{\operatorname{argmax}} \frac{\alpha^\intercal K_a K_b \beta}{\sqrt{\alpha^\intercal K_a^2 \alpha}\sqrt{\beta^\intercal K_b^2 \beta}} \tag{3.34}$$

Hardoon et al. (2004) observed that KCCA frequently suffers over-fitting, especially when dealing with high-dimensional data, and applied regularisation to control the over-fitting:

$$\underset{\alpha,\beta}{\operatorname{argmax}} \frac{\alpha^\intercal K_a K_b \beta}{\sqrt{\left(\alpha^\intercal K_a^2 \alpha + \kappa \left\|w_a\right\|^2\right)\left(\beta^\intercal K_b^2 \beta + \kappa \left\|w_b\right\|^2\right))}} \tag{3.35}$$

It follows that:

$$\underset{\alpha,\beta}{\operatorname{argmax}} \frac{\alpha^\intercal K_a K_b \beta}{\sqrt{\left(\alpha^\intercal K_a^2 \alpha + \kappa \alpha' K_a \alpha\right)\left(\beta^\intercal K_b^2 \beta + \kappa \beta^\intercal K_b \beta\right)}} \tag{3.36}$$

where $\kappa$ denotes a penalty term that can control regularisation strength. Similarly to CCA, since the problem is not affected by scaling of $\alpha$ and $\beta$, it can be reformulated as a maximisation of the numerator subject to the following constraints:

$$\left(\alpha^\intercal K_a^2 \alpha + \kappa \alpha^\intercal K_a \alpha\right) = 1 \tag{3.37}$$

$$\left(\beta^\intercal K_b^2 \beta + \kappa \beta^\intercal K_b \beta\right) = 1 \tag{3.38}$$

Through the Lagrange formulation, this leads to a standard eigenproblem:

$$(K_a + \kappa I)^{-1} K_b (K_b + \kappa I)^{-1} K_a \alpha = \lambda^2 \alpha \tag{3.39}$$

The eigenvalues of Equation 3.39 are the canonical correlations and the eigenvectors can be used to calculate the projections. The problem can be solved in different ways; however, an effective algorithm (PGSO) proposed by Hardoon et al. (2004) is chosen and I implement it in Python.

### 3.5.2 KCCA-based Cross-lingual Word Embedding

Let $\tilde{X}_s \in \mathbb{R}^{\tilde{N}_s \times d_s}$ and $\tilde{X}_t \in \mathbb{R}^{\tilde{N}_t \times d_t}$ be the monolingual word embeddings matrices of source and target language vocabularies. In supervised scenario, a set of word embeddings of translation pairs (i.e. a dictionary) is given: Let $X_s \in \mathbb{R}^{N_s \times d_s}$ contain a subset of word embeddings from $\tilde{X}_s$, and $X_t \in \mathbb{R}^{N_t \times d_t}$ contain their corresponding translation word embeddings from $\tilde{X}_t$. A pair of rows with the same index from each matrix represents a translation pair. The proposed approach can be broken down into three steps: pre-processing (described in Section 3.4.1), KCCA-projection and re-weighting (described in Section 3.4.3) based on canonical correlations.

Given word embedding matrices $X_s$ and $X_t$, as defined above, the KCCA implementation described in Section 3.5.1 is adopted in the following way. The projection matrix are learnt using Equation 3.40:

$$\alpha, \beta, \rho = KCCA(X_s, X_t) \tag{3.40}$$

In Equation 3.40, $\alpha$ and $\beta$ are components of projection matrix described in Equation 3.31 and Equation 3.32. In addition, $\rho$ are the canonical correlations corresponding to each of the projection directions. Radial basis function (RBF) kernel is used and the value of parameter $\gamma$ is tuned through cross-validation.

Given $\alpha$ and $\beta$ calculated by KCCA, the vocabulary word embedding matrices $\tilde{X}_s$ and $\tilde{X}_t$ are projected into the shared space:

$$\tilde{X}_s^* = K(\tilde{X}_s, X_s^T)\alpha \tag{3.41}$$

$$\tilde{X}_t^* = K(\tilde{X}_t, X_t^T)\beta \tag{3.42}$$

### 3.5.3 Deep Canonical Correlation Analysis

An alternative non-linear transformation is Deep Canonical Correlation Analysis (DCCA), firstly proposed by Andrew et al. (2013). Similar to KCCA, DCCA is proposed to learn non-linear mappings. In this section, I briefly introduce Deep Canonical Correlation Analysis (DCCA).

DCCA computes two new multivariate representations by feeding input features into a feed-forward neural network, which enables non-linear transformations. For simplicity, I only consider a two-view analysis with two inputs $X_1$ and $X_2$. An $n$ layer neural network with $c_i$ ($i \in \mathbb{R}^n$) units in each layer is used. Assume the output layer has $c_o$ units. Consider an instance of $X_1$ denoted by $x_i \in \mathbb{R}^{N_1}$, the first hidden layer's output $h_1 \in \mathbb{R}^{c_1}$ for $x_i$ is:

$$h_1 = f(W_1^1 x_i + b_1^1) \tag{3.43}$$

where $W_1^1 \in \mathbb{R}^{c_1 \times n_1}$ denotes the weight matrix between the first layer and the hidden layer, $b_1^1$ denotes the corresponding bias, and $f : \mathbb{R} \to \mathbb{R}$ denotes a non-linear activate function. The $h_1$ is then used to compute the next hidden state and so on until the final output of the output layer is computed:

$$o_1 = f(W_n^1 h_{n-1} + b_n^1) \tag{3.44}$$

where $o_1 \in \mathbb{R}^{c_{o1}}$ denotes the final non-linear projection of $x_i$. Also DCCA compute the final non-linear projection of $x_2$ as $o_2 \in \mathbb{R}^{c_{o2}}$ in the same way. Based on this process, those two neural networks are defined as two non-linear transformation $F_1(X_1; \theta_1)$ and $F_2(X_2; \theta_2)$ which aims to map $X_1$ and $X_2$ into a shared space. The goal of DCCA is to train those neural networks so that the Pearson correlation of the final projections of $X_1$ and $X_2$ are mutually maximised. Therefore, the loss function $L$ is defined as:

$$L(\theta_1, \theta_2) = \text{corr}(F_1(X_1; \theta_1), F_2(X_2; \theta_2)) \tag{3.45}$$

As discussed in Equation 3.16 to 3.20, the sum of correlation is equal to the sum of the top $k$ singular values of matrix $T = USV^\intercal$. In DCCA, if the output layer unit number $o = k$, A Pearson correlation can be obtained:

$$\text{corr}\,(H_1, H_2) = \|T\|_{tr} = \text{tr}(T^\intercal T)^{\frac{1}{2}} \tag{3.46}$$

The Pearson correlation denotes the total loss function. $H_1$ and $H_2$ denotes the centered

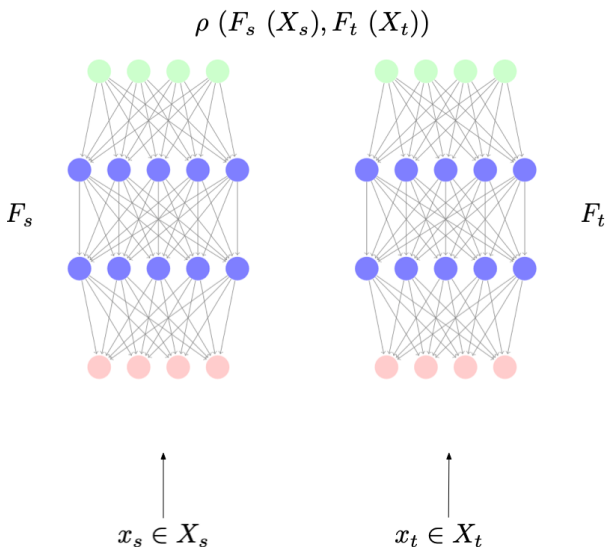



**Figure 20:** DCCA-based cross-lingual word embeddings.

data matrices. The gradient of DCCA model is expressed as Equation 3.47:

$$\frac{\partial \text{corr}\ (H_1, H_2)}{\partial H_1} = \frac{1}{m-1}(2\nabla_{11}\overline{H}_1 + \nabla_{12}\overline{H}_2) \tag{3.47}$$

The $\nabla_{12}$ , $\nabla_{11}$ and $\nabla_{22}$ can be calculated using Equation 3.48 to 3.50

$$\nabla_{12} = \hat{C}_{11}^{-\frac{1}{2}} U V^\intercal \hat{C}_{22}^{-\frac{1}{2}} \tag{3.48}$$

$$\nabla_{11} = -\frac{1}{2}\hat{C}_{11}^{-\frac{1}{2}} U S U^\intercal \hat{C}_{11}^{-\frac{1}{2}} \tag{3.49}$$

$$\nabla_{22} = -\frac{1}{2}\hat{C}_{22}^{-\frac{1}{2}} U S U^\intercal \hat{C}_{22}^{-\frac{1}{2}} \tag{3.50}$$

Similar to Section 3.3.3, the $\hat{H}_1$ and $\hat{H}_2$ denote the regularised matrices upon centred matrices.

### 3.5.4 DCCA-based Cross-lingual Word Embedding

As discussed before, given two dictionary word embedding matrices $X_s$ and $X_t$. The paired word vectors can be seen as training data for DCCA. Figure 20 summarises the DCCA-based cross-lingual word embedding. Word pair $(x_s^i, x_t^i) \in (X_s, X_t)$ passes through two separate neural networks $F_s(\theta_s)$ and $F_t(\theta_t)$, where $\theta_s$ and $\theta_t$ are parameters of the corresponding neural networks. The output representations are the input data's non-linear transformations (learnt by the neural networks). Subsequently, those outputs are mutually

maximised:

$$\theta_s, \theta_t = \underset{\theta_s,\theta_t}{\text{argmax}}(\text{corr}(F_s(X_s;\theta_s), F_t(X_t,\theta_t))) \tag{3.51}$$

After training process, the two trained neural networks map the two vocabulary word embedding matrices $\tilde{X}_s$ and $\tilde{X}_t$ into a shared vector space:

$$\tilde{X}_s^* = F_s(\tilde{X}_s;\theta_s) \tag{3.52}$$

$$\tilde{X}_t^* = F_t(\tilde{X}_t,\theta_t) \tag{3.53}$$

## 3.6 Experiments

Experiments described in this section investigate whether non-linear mapping-based approaches are better than linear mapping-based approaches in learning cross-lingual word embeddings. These experiments evaluate CCA-, DCCA- and KCCA-mapped word embeddings using the accuracy of a downstream task - word translation - as a proxy measure of "better". This is a standard evaluation technique in cross-lingual word embedding literature. Experimental results are also compared with other linear-mapping-based approaches.

### 3.6.1 Datasets

English-Italian (En-It) dataset provided by Dinu and Baroni (2015) is a commonly-used dataset to learn cross-lingual word embeddings. Artetxe et al. (2017) further extend this dataset to include English-German (En-De), English-Spanish (En-Es) and English-Finnish (En-Fi) datasets. Each dataset includes 20,000 300-dimensional monolingual word embeddings trained using word2vec, along with a bilingual dictionary with a standard split into a training and a test set. Those dictionaries were obtained from OPUS and include 5000 most frequent word pairs as the training set and 1500 randomly picked word pairs evenly distributed in 5 frequency bins (Tiedemann, 2012). In terms of monolingual word embeddings, the English training corpora consists of 2.8 billion words and included ukWaC, Wikipedia and BNC. The Italian training corpora included 1.8 billion words for itWaC. German training corpora used SdeWac with 0.9 billion words, and Finnish training corpora used the Finnish WMT 2016 dataset (Common Crawl). The Spanish word vectors were obtained by training WMT News Crawl 07 - 12, consisting of 386 million words.

### 3.6.2 A New Word Translation Corpora: English-Chinese Dataset

I hypothesised that non-linear mapping can better describe the relationship between word embeddings of languages in different language families. In order to test this hypothesis, I extended the dataset to include a pair of languages from different families: English paired with Chinese, a Sino-Tibetan language. I trained the Chinese word embeddings on a 1.5 billion word subset of the WMT 2018 Common Crawl corpora. Unlike Western language families, Chinese tokenisation needs a specific process to extract words from sentences. I adopt the solution from an open project: Jieba3[1]. The word embeddings are trained using the same configuration as Dinu and Baroni (2015). As for the dictionary, the English-Chinese dictionary provided by Lample et al. (2018) is used, consisting of 8728 training word pairs and 2230 test word pairs. After removing out of vocabulary (OOV) words from the dictionary, it is left with 8239 training word pairs and 1964 test word pairs. I provide it as an open-source dataset[2].

### 3.6.3 Evaluation

The accuracy of the word translation downstream task is used as a proxy to evaluate the learnt cross-lingual word embeddings, in a similar fashion to other previous work in this area (Artetxe et al., 2018b,a; Smith et al., 2017). The test word embedding matrices are projected to the shared vector space using the learnt projection matrices during the evaluation process. Then, a retrieval approach is used to find the nearest neighbour of each source word embedding in the target language vocabulary word embedding matrix. The word translation accuracy is defined as the percentage of correct matches from source to target words in the test set.

There are many retrieval methods of finding nearest neighbours, like simply using a distance measure (i.e. cosine similarity or Euclidean distance) or inverted softmax (Smith et al., 2017). However, most retrieval methods suffer from *hubness*: **Some vectors, dubbed *hubs*, are the nearest neighbours of many other points, while others (so called *anti-hubs*) are not nearest neighbours of any points**. This issue has been observed in many applications of information retrieval (i.e. image matching, translating words). The hubness issue has a negative impact on downstream tasks such as classification (Radovanović et al., 2009). Cross-domain Similarity Local Scaling (CSLS) is

[1] https://github.com/fxsjy/jieba
[2] https://gitlab.com/zjw1990/kclwe

an advanced retrieval method for mitigating problems caused by hubness (Lample et al., 2018).

In this investigation, I choose CSLS as the retrieval method in word translation task. Given a source language word embedding vector mapped into the shared vector space $x_s^*$, CSLS method finds $x_s^*$'s top K nearest neighbours amongst target word embeddings mapped into the shared vector space using cosine similarity and averages them:

$$r_t = \frac{1}{K}\sum_{i=1}^{K} S_c(x_s^*, x_t^*[i]) \tag{3.54}$$

where $x_t^*$ is one of the top K nearest neighbours amongst target language mapped word embedings. Similarly, $r_s$ is defined as the mean similarity of a mapped target word vector to its top K nearest neighbours in the mapped word embeddings of the source language. Then the CSLS score is defined to measure two mapped embeddings' distance:

$$CSLS(x_s^*, x_t^*) = 2S_c(x_s^*, x_t^*) - r_t - r_s \tag{3.55}$$

where $S_c$ is cosine similarity:

$$S_c(x_s, x_t) = \frac{\sum_{i=1}^{d} x_s^i x_t^i}{\sqrt{\sum_{i=1}^{d}(x_s^i)^2}\sqrt{\sum_{i=1}^{d}(x_t^i)^2}} \tag{3.56}$$

### 3.6.4 Experimental Setup

Two different kernel functions are evaluated for the proposed cross-lingual word embeddings: RBF kernel and polynomial kernel. Given two samples $(x, y)$, the dual RBF kernel in input space is defined as:

$$K(x, y) = exp(-\frac{\|x-y\|^2}{2\sigma^2}) \tag{3.57}$$

For simplicity, a parameter $\gamma = \frac{1}{2\sigma^2}$ is defined. Substituting into Equation 3.57 we get:

$$K(x, y) = exp(-\gamma\|x-y\|^2) \tag{3.58}$$

This parameter $\gamma$ is tuned in the range [0, 1.5]. The weight for re-weighting is tuned in the range [0, 1] and the regularisation term $\kappa$ is tuned in the range [0, 1]. For CCA, the output dimension is tuned in the range from 150 to 300.

The polynomial kernel is defined as:

$$K(x, y) = (x^T y + c)^d \tag{3.59}$$

In practice, the polynomial kernel is hard to tune. When $d > 3$ the computational complexity grows proportionally. Therefore, I only tune parameter $c$ in the experiments and tune $d$ in (1,2,3).

DCCA-based cross-lingual word embedding model proposed by Lu et al. (2015) is adopted. However, their published model uses older count-based word representations, and did not work very well with the distributed word embeddings used with other methods. I adapt the model by tuning a new parameter set with distributed word embeddings used as model input. I use two feed-forward neural networks with ReLU activation functions in hidden layers. The hidden layer size is tuned amongst a set of values $128, 256, 512, 1024, 2048, 4096$ and the depth parameter is tuned amongst $1, 2, 3, 4$. Parameters are tuned separately for each language. As for optimisation, the stochastic gradient descent (SGD) is used. The regularisation terms $r_x$ and $r_y$ are tuned in the range [1e-9, 1e-5]. All tuning is performed on the training set using 5-fold cross-validation.

CCA, as described in Section 3.3.3, is implemented. The final dimension is tuned amongst $100, 150, 200, 250, 300$. For the re-weighting process, the re-weighting parameter is tuned in range [0, 1].

For SVD (Orthogonal), the open source software Vecmap[3] is adopted, which also provides the standard pipeline for mapping-based methods. The experiments were implemented in Python3 using an open source library Numpy[4]. The experiments were carried out on AMD Threadripper 1950x 16-core CPU with 128Gb RAM.

## 3.7 Results and Analysis

### 3.7.1 Analysis Between Non-Linear Methods and Linear Methods

Table 3.2 shows that KCCA outperforms CCA- and SVD-based methods, which reveals the effectiveness of the non-linear method. KCCA outperforms CCA, giving a 6.29, 12.58, 10.66, 11.73 and 19.96 percentage points improvement in English-Italian, English-German, English-Spanish, English-Finnish and English-Chinese datasets respectively. KCCA also outperforms SVD-based method, providing a 1.1, 2.9, 0.7, 2.4 and 3.4 percentage points improvement word translation accuracy.

Also, the experiments show that DCCA-based non-linear mapping is better than CCA-based linear mapping approach. The improvement in word translation task demonstrates

[3] https://github.com/artetxem/vecmap
[4] https://numpy.org/

**Table 3.2:** A comparison of linear mapping with non-linear mapping using of word translation accuracy %.

| Method | **En-It** | **En-De** | **En-Es** | **En-Fi** | **En-Zh** |
|---|---|---|---|---|---|
| Linear | | | | | |
| CCA | 42.1 | 37.6 | 28.2 | 25.7 | 32.6 |
| SVD (Orthognal) | 47.3 | 47.2 | 38.2 | 35.0 | 49.2 |
| Non-linear | | | | | |
| DCCA | 43.5 | 43.1 | 34.9 | 25.3 | 45.3 |
| KCCA | **48.4** | **50.1** | **38.9** | **37.4** | **52.6** |

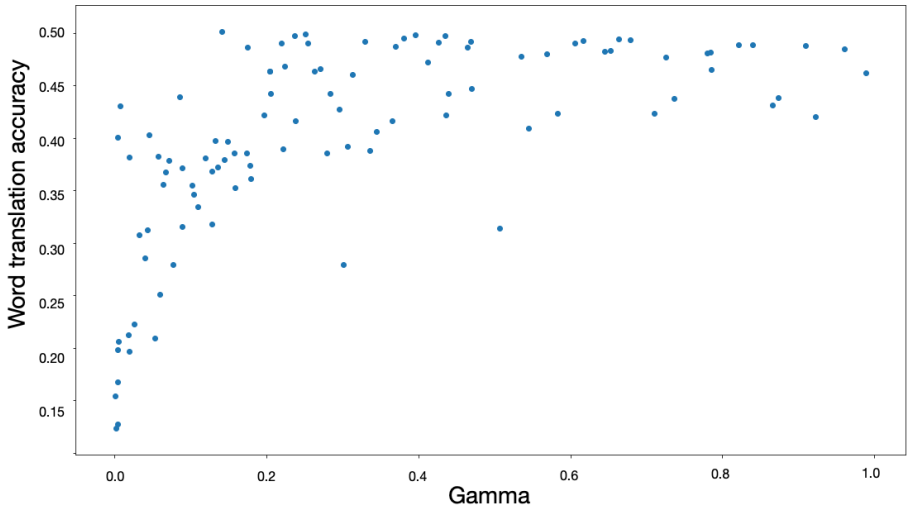
**Table 3.3:** A comparison of KCCA-based mapping with existing methods using word translation accuracy (%). All existing results are from original papers, except results marked with a $^\star$, which were produced by me using authors' original implementation.

| Method | **En-It** | **En-De** | **En-Es** | **En-Fi** |
|---|---|---|---|---|
| Mikolov et al. (2013a) | 34.93 | 35.00 | 27.73 | 25.91 |
| Faruqui and Dyer (2014) | 38.40 | 37.13 | 26.80 | 27.60 |
| Artetxe et al. (2016) | 39.27 | 41.87 | 31.40 | 30.62 |
| Smith et al. (2017) | 44.53 | 43.33 | 35.13 | 29.42 |
| Artetxe et al. (2018a) (nn) | 45.27 | 44.27 | 36.60 | 32.94 |
| Artetxe et al. (2018a) (CSLS) | 47.33$^\star$ | 47.20$^\star$ | 38.20$^\star$ | 34.97$^\star$ |
| KCCA | **48.33** | **50.13** | **38.86** | **37.43** |

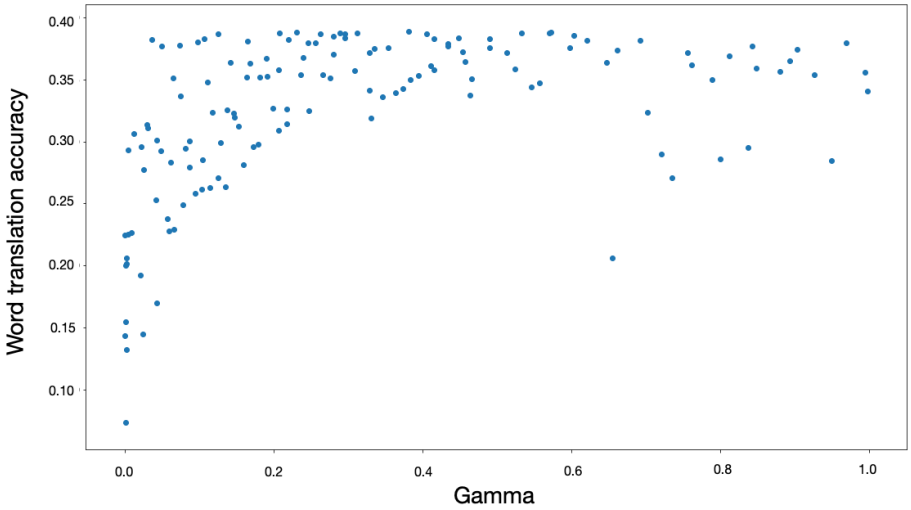
that the use of a non-linear transformation outperforms a linear transformation, and serves as further evidence to confirm the hypothesis that relationship between word embeddings of different languages has a strong non-linear component.

### 3.7.2 Analysis Between Non-linear Methods

It is also worth noting that KCCA outperforms DCCA on all datasets, specifically it performs better by 4.3, 7, 4, 12.15 and 7.22 percentage points in English-Italian, English-German, English-Spanish, English-Finnish and English-Chinese respectively. This result suggests that the neural network may be struggling to learn cross-lingual word embeddings from a fairly small training set, where as kernel-based methods can better deal with data sparsity. Table 3.3 shows a comparison between the proposed approach and previous popular works, including supervised, semi-supervised and fully unsupervised methods. The best result of the supervised approaches is provided by Artetxe et al. (2018a); however, their retrieval approach is inverted softmax (Smith et al., 2017), which is not a fair comparison with CSLS. Therefore, I also reproduce their result using CSLS and report it in Table 3.3. It can be observed that my proposed framework gets the best result among all supervised settings. Also, other than a close result in the English-Italian dataset, my proposed approach achieves the best results in all language pairs compared with the un-




**(a)** En-De dataset



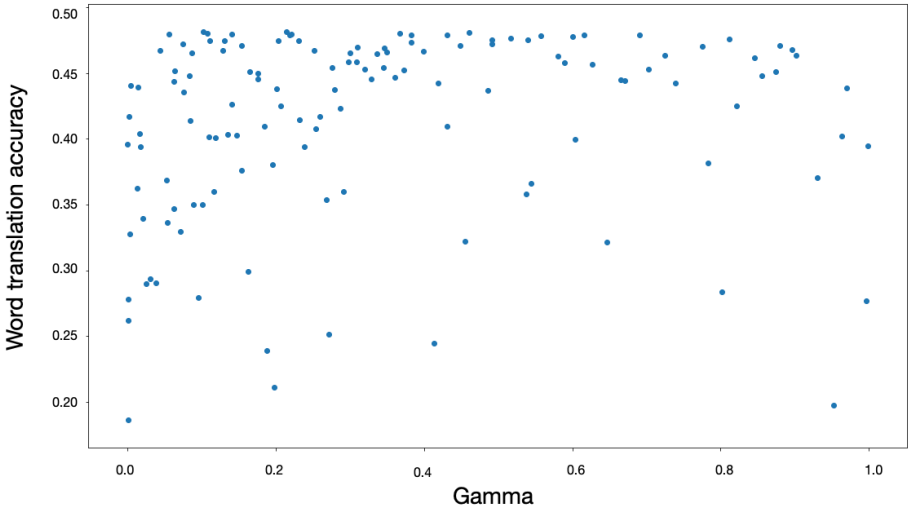

**(b)** En-Es dataset

**Figure 21:** Word translation accuracy of test set dictionary versus different values of gamma on En-De and En-Es dataset. x-axis is gamma, and y-axis is word translation accuracy of the test.

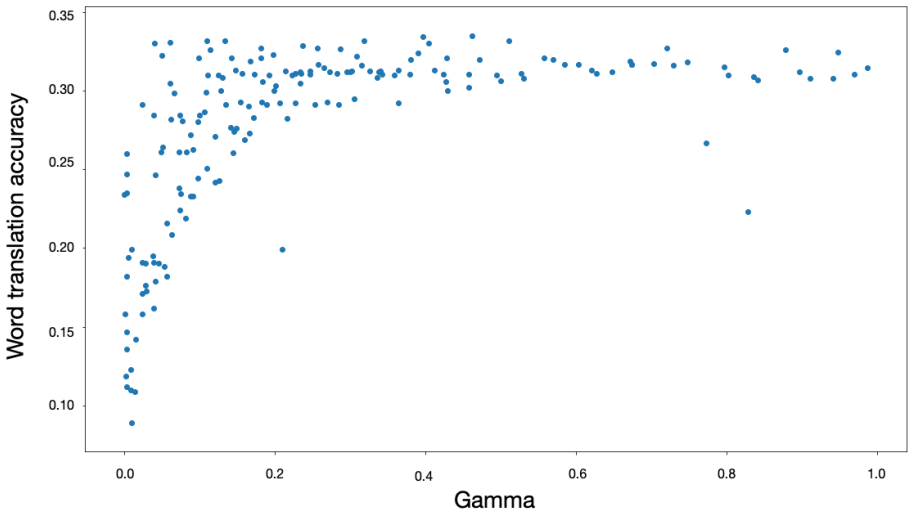
supervised setting proposed by Artetxe et al. (2018b), which is also the state-of-the-art of existing cross-lingual word embedding approaches.

### 3.7.3 Further Analysis for KCCA

**Analysis of KCCA-Mapping process**: In my experiments, $\gamma$ parameter is tuned using a grid search. Figure 21 illustrates the effect of $\gamma$: when $\gamma > 0.3$, it has no effect on the result. However, as $\gamma$ decreases below 0.3, the performance of the produced cross-lingual word embeddings is more likely to be worse - the results seem to be much more sensitive to the exact value of $\gamma$ at low values than at higher levels.

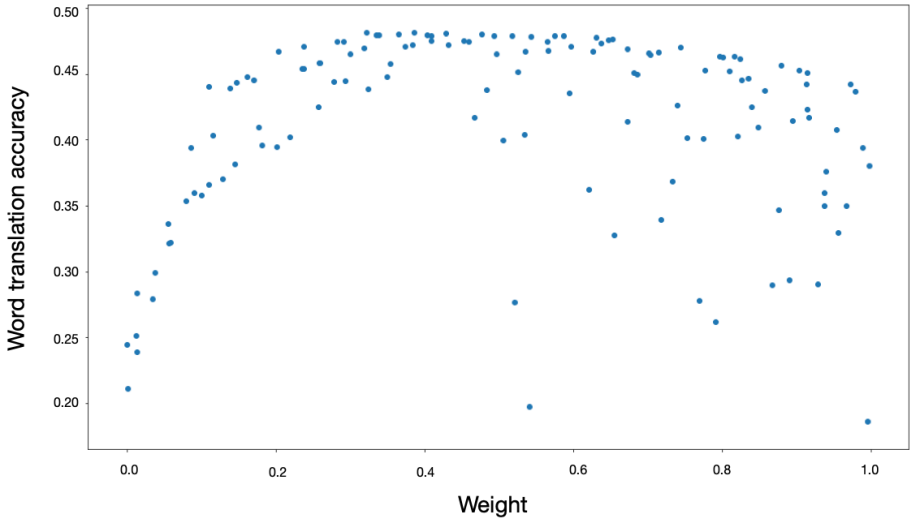



**(a)** En-It dataset.

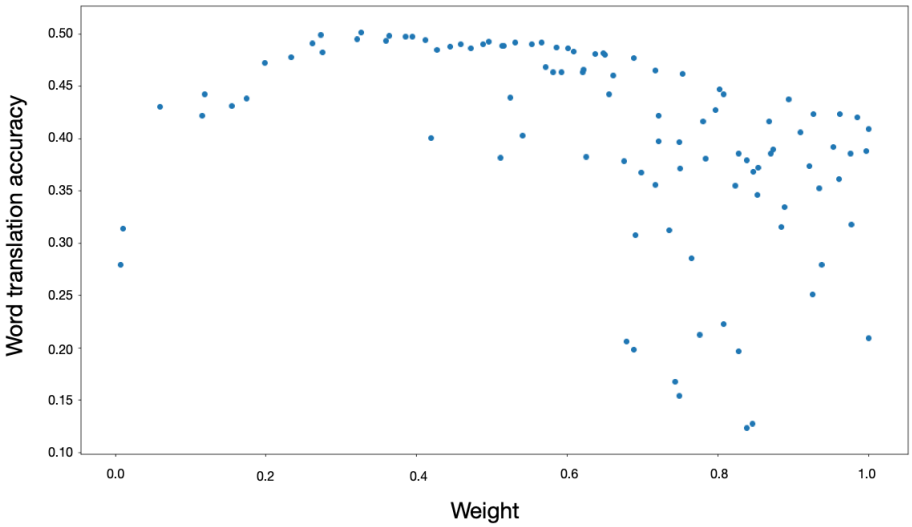



**(b)** En-Fi dataset.

**Figure 22:** Word translation accuracy of test set dictionary versus different values of gamma on En-It and En-Fi dataset. x-axis is gamma, and y-axis is word translation accuracy of the test.




**(a)** En-It dataset.

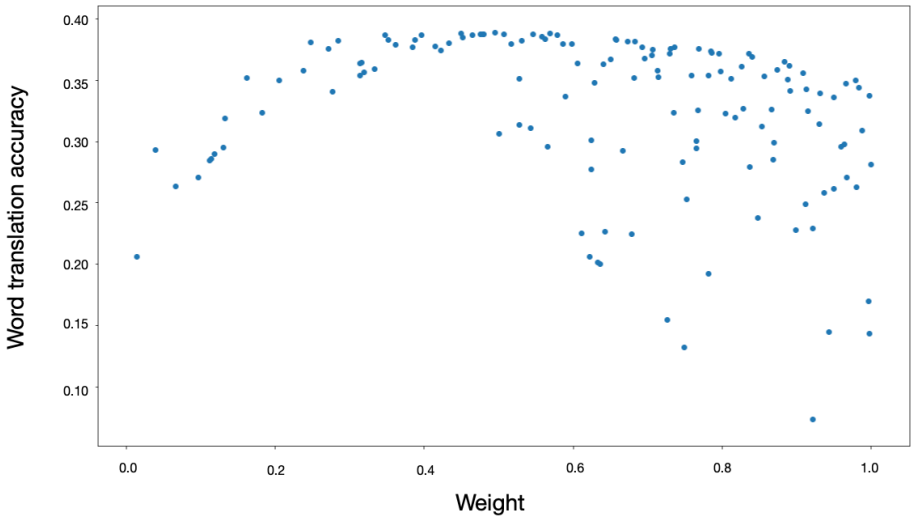



**(b)** En-De dataset.

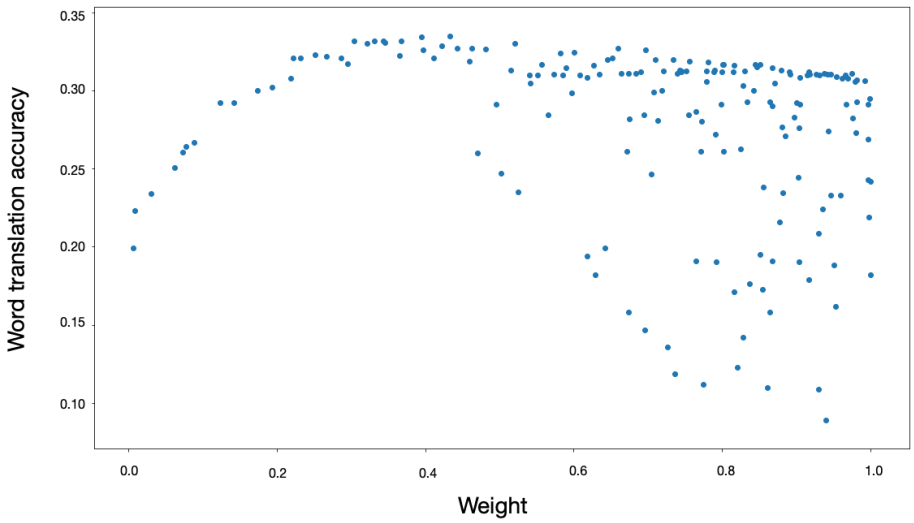
**Figure 23:** Word translation accuracy of En-It and En-De test set dictionary versus different values of weight $w$. x-axis is the weight parameter, and y-axis is word translation accuracy of the test.

**Analysis of the re-weighting process:** The re-weighting process is an important post projection step to produce better cross-lingual word embeddings. Some previous analysis showed that re-weighting outperforms dimensionality reduction (Artetxe et al., 2018a). From Figure 24, a clear trend can be observed that the best results tend to be obtained when $w$ is around 0.4 for all language pairs.

**Analysis of the dimensionality reduction process** Due to limited computational resources, I was able to investigate the effect of dimensionality reduction only on En-Es dataset. The results are shown in Figures 25 and 26. Figure 25 shows that the best value of the final dimension is around 300 - for values of d slightly higher or lower, the accuracy




**(a)** En-Es dataset.




**(b)** En-Fi dataset.

**Figure 24:** Word translation accuracy of En-Es and En-Fi test set dictionary versus different values of weight $w$. x-axis is the weight parameter, and y-axis is word translation accuracy of the test.

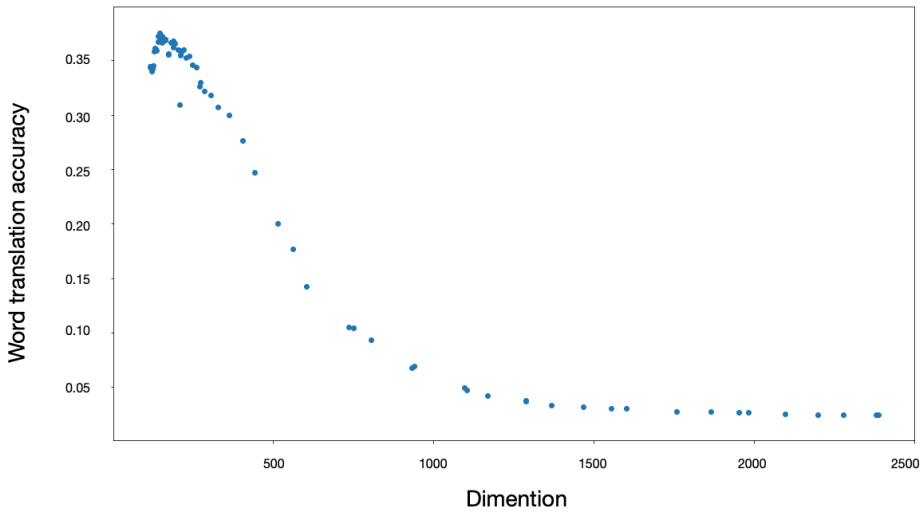



**Figure 25:** Word translation accuracy of test set of En-Es dataset applying different dimensions.

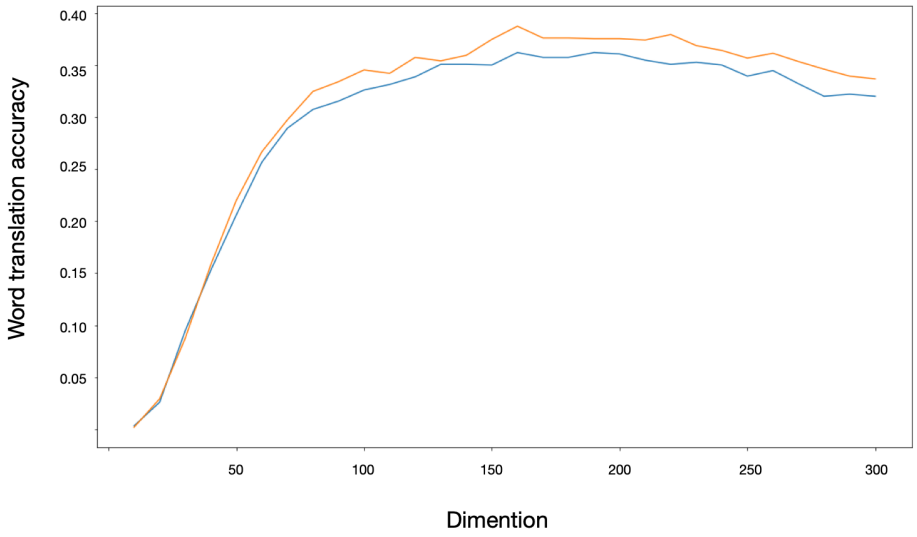



**Figure 26:** Word translation accuracy comparison of CCA and KCCA when they have the same dimension. The blue line is CCA and the red line represents KCCA.

sharply decreases. Figure 26 shows that KCCA outperforms CCA for all values of the final projection dimension d, which serves as an additional confirmation of my hypothesis that non-linearity can improve the performance of the cross-lingual word embeddings.

### 3.7.4 Ratio R

The results also lead to an interesting question: what kind of words are correctly translated when non-linear mapping-based approaches are used. I take the En-Fi dataset as an example. Figure 29 illustrates that both CCA and KCCA correctly translate the same 320 English words (yellow points), KCCA can correctly translate 213 words that CCA fails

**Table 3.4:** A comparison of correct translations by CCA- and KCCA-based methods.

| Method | En-It | En-De | En-Es | En-Fi | En-Zh |
|---|---|---|---|---|---|
| CCA only correct | 60 | 46 | 41 | 46 | 25 |
| Both Correct | 572 | 517 | 382 | 320 | 395 |
| KCCA only correct | 154 | 235 | 200 | 213 | 282 |
| Ratio, R | 24.4% | 41.7% | 47.3% | **58.2%** | **67.1%** |

to translate (orange points); however, CCA can translate only 46 words that KCCA fails on (green points). I believe the words that are correctly translated by KCCA but incorrectly translated by CCA have a higher possibility of exhibiting non-linear relationships; The number of those words is denoted as N. The word pairs correctly translated by CCA have a higher possibility of sharing a linear relationship. The count of those word pairs is denoted as L. Then define a ratio R:

$$R = \frac{N}{L} \tag{3.60}$$

as a proxy measure to evaluate whether a relationship between word embeddings of two languages has a more linear or non-linear tendency; Table 3.4 provides the results. It shows that Italian words have the highest possibility to share linear relationships with English. Most German and Spanish words can map to English words with linear projections, but a considerable number of words can not match with the projections. Non-linear relationships have a significant impact on Finnish-English word pairs and Chinese-English word pairs. In my opinion, this is because different languages have different grammars, which leads to varying contexts for words with similar meanings in both languages. However, the better result on the En-De dataset seems to be not following this trend; it has a lower R, but non-linearity based methods can learn better cross-lingual word embeddings than the linear methods. The most important factor is the limited data. The trend for R will be more clear clear if more language pairs are included in the experiment and this will be part of my future works.

## 3.8 Conclusion

In this chapter, I investigated whether non-linear relationships exist between the word vector representations of different languages. Additionally, I posited that non-linear mapping methods could produce better quality cross-lingual representations.

To confirm the proposed hypothesis, I investigated two non-linear methods: KCCA- and DCCA- based cross-lingual word embedding approaches. I further described a pipeline

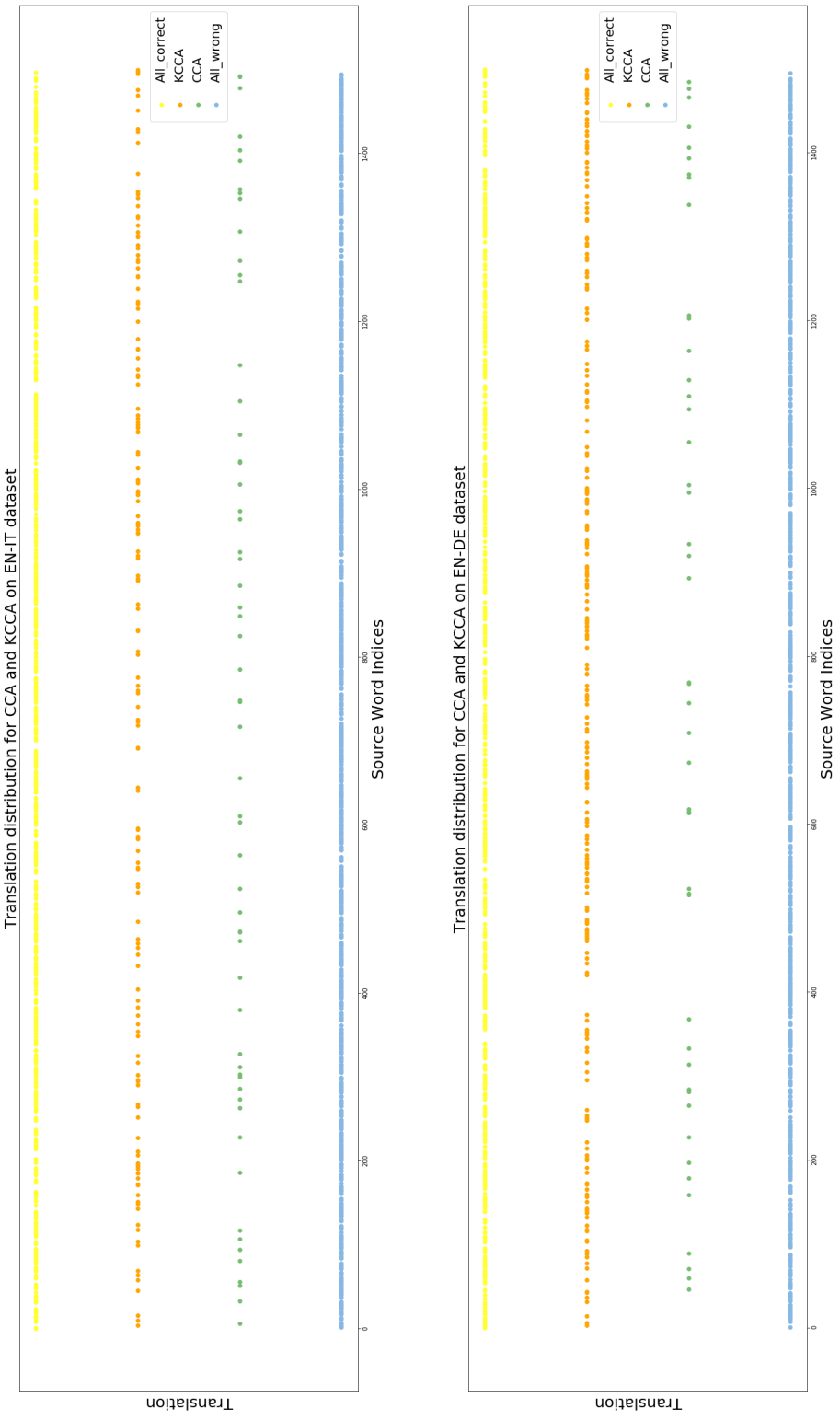



**Figure 27:** The translation result on the En-It and En-De test sets. x axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only.

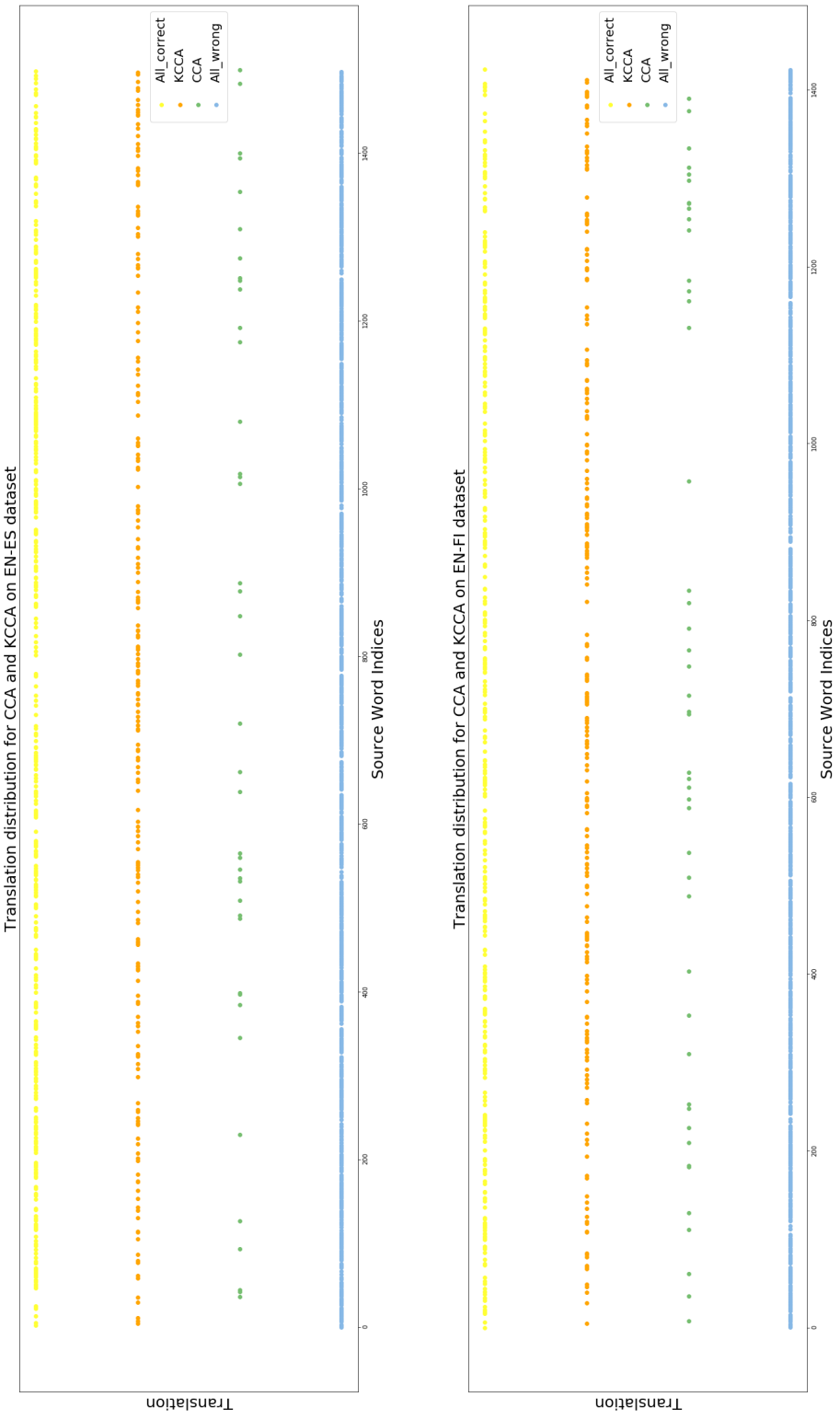



**Figure 28:** The translation result on the En-Es and En-Fi test sets. x axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only.

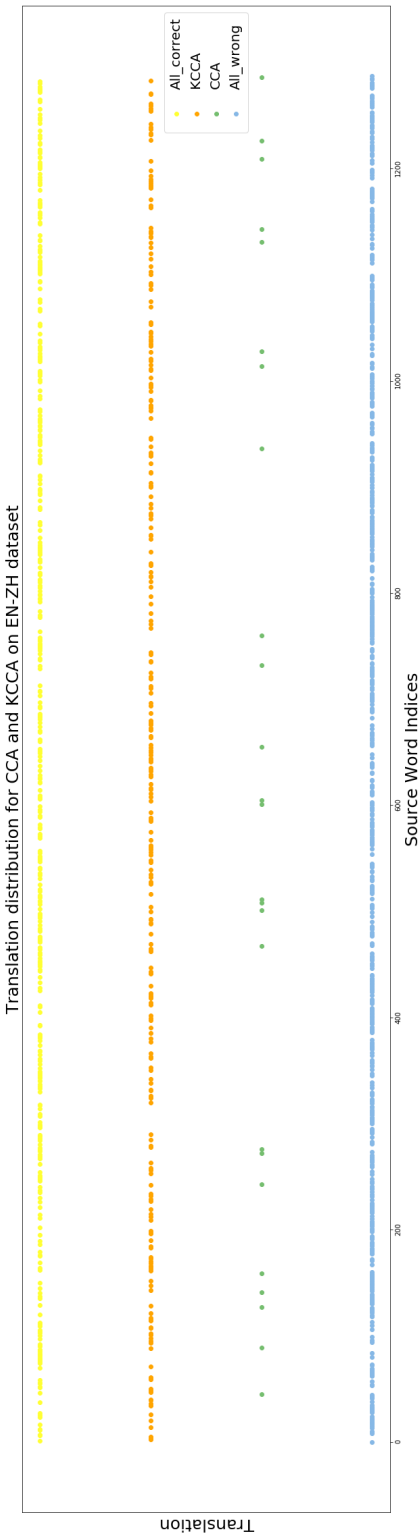



**Figure 29:** The translation result on the En-Zh test sets. The x axis represents the indices of English words. Yellow points indicates English words correctly translated by both CCA and KCCA. Orange points denote words correctly translated by KCCA only. Green points denote English words correctly translated by CCA only. Blue points denote incorrect translation by both CCA and KCCA. It can be observed that many more words are translated correctly by KCCA-only than CCA-only.

of cross-lingual word embeddings and adapt my proposed approach to it.

The use of KCCA for mapping obtained the best result among all methods, confirming my hypothesis that non-linearity could better describe the relationship between two languages, especially those not in the same language family. Also, I further measure the relationships with a ratio R: the language pairs with a larger R indicates they have a different relationship. Also, I provide state-of-the-art results on all language pairs.

CHAPTER 4

# Learning Unsupervised Cross-Lingual Word Embeddings with Non-Linear Mapping

## 4.1 Introduction

In recent years, there has been an increased interest in learning cross-lingual word embeddings. Those approaches can be generally classified into three types. These are the: mapping-based approach, the joint model-based approach and the pseudo-bi-lingual-based approach (Ruder et al., 2019; Adams et al., 2017; Kočiský et al., 2014; Mikolov et al., 2013a).

*The mapping-based approach* is the most widely used method because of its simplicity and the ease of use. It aims to map monolingual word embeddings into a shared vector space using a bi-lingual dictionary (also called word alignment) (Ruder et al., 2019; Artetxe et al., 2018a).

A serious issue with the mapping-based approach, however, is its requirement for an extensive dictionary, which is hard to obtain for low-resource languages (Artetxe et al., 2018b; Lample et al., 2018; Artetxe et al., 2017). As illustrated in Table 4.1, there is a sharp decline in the translation accuracy when the dictionary becomes smaller. Additionally, compared to the high resource language pairs (e.g. English-Italian), the low-resource

**Table 4.1:** The word translation accuracy % of the En-It (high-resource) and the En-Fi (low-resource) dataset with orthogonal mapping. The retrieval method is cosine similarity based nearest neighbour.

| Dictionary size | 100 | 500 | 1000 | 2000 | 4000 | 5000 |
|---|---|---|---|---|---|---|
| En-It | 0.0 | 2.1 | 28.3 | 37.9 | 41.8 | 43.1 |
| En-Fi | 0.0 | 0.0 | 16.6 | 27.0 | 31.5 | 32.0 |

language pairs (e.g. English-Finnish) are more sensitive to the size of the training dictionary. The word translation accuracy declines even faster than that of the rich resource language pairs. Therefore, a high-quality bi-lingual dictionary is useful to learn cross-lingual word embeddings in low-resource languages.

A variety of studies have attempted to address the inadequate bi-lingual dictionary issue for low resource languages (Artetxe et al., 2018b; Lample et al., 2018; Artetxe et al., 2017). Most of these studies have only focused on linear mapping-based methods. However, the low-resource languages (e.g. Turkish, Finnish) and high-resource languages (e.g. English, Spanish) are normally in different language families. For instance, heavily used languages like English, German and Italian belong to the Indo-European language family, while a typical low-resource language like Finnish belongs to the Uralic language family. Many eastern languages like Chinese and Karenic languages belong to the Sino-Tibetan language family (Thurgood and LaPolla, 2016). Members of the same language family are likely to share similar syntax rules and alphabet. Those that are in different language families are likely to share different syntax rules and alphabet. Therefore, the sentences in the different language family tend to have similar meaning for a specific word, making their word2vec word representations share geometric arrangements.

Thus, the word embeddings of a low-resource language are more likely to have a non-linear relationship with that of high-resource languages. As discussed in Chapter 3, the non-linear method can thus better describe such a relationship.

In this chapter, I discuss non-linear mapping in unsupervised cross-lingual word embeddings. I extend the proposed non-linear mapping based cross-lingual word embedding to an unsupervised scenario. Without any manually created dictionary, an unsupervised dictionary selection process is proposed to generate the dictionary automatically. The proposed model can learn better cross-lingual word embeddings employing the monolingual corpora, which is helpful for low-resource languages. Extensive experiments are provided, and the results from the proposed approach achieve the state-of-the-art.

## 4.2 Related Works

### 4.2.1 Supervised Cross-Lingual Word Embedding Approaches

Mikolov et al. (2013a) were the first to propose the supervised cross-lingual word embedding approach aiming to learn a mapping matrix by minimising the Euclidean distance between the dictionary word embedding matrices $W_s$ and $W_t$ (See Section 3.3). Follow-

ing on from this, Xing et al. (2015) found that the combination of word embeddings with unit length, as well as constraining the mapping to be orthogonal, help to improve the performance of cross-lingual word embeddings. Artetxe et al. (2016) further demonstrated that the orthogonal mapping constraint is a preserve of monolingual invariance. Artetxe et al. (2018a) formulate previous works and provide a general five-step pipeline for supervised cross-lingual word embeddings.

An alternative objective is to maximise the cross-correlations between words in dictionaries, which can be learned by Canonical Correlation Analysis (CCA). Faruqui and Dyer (2014) were the first to apply CCA to construct cross-lingual word representations from two sets of monolingual ones from two different languages, and demonstrated that their use improves a number of tasks over monolingual representations, such as finding word similarity and semantic as well as syntactic relations. They attribute the improved performance to the idea that shared representation is able to incorporate lexicon-semantic information from both languages. Ammar et al. (2016) extend this work to a multi-lingual scenario, which is able to project word vectors from multiple languages into a same shared word embedding space.

Previous works entail linear-mapping-based methods, while Zhao and Gilman (2020) extend those approaches to a non-linear-mapping-based method. However, supervised approaches require a large dictionary, which is hard to obtain in most language pairs.

### 4.2.2 Unsupervised Mapping-based Cross-Lingual Word Embedding Approaches

Scarce availability of large dictionaries for many language pairs has spurred an interest in alternative unsupervised approaches. Bootstrapping was proposed as one solution. The bootstrapping-based unsupervised approach aims to bootstrap a high-quality bi-lingual word embedding space from a tiny seed dictionary. Bootstrapping has been applied to traditional count-based word vector space models to construct cross-lingual word embeddings in a weakly supervised fashion using only a small dictionary (Artetxe et al., 2017; Peirsman and Padó, 2010).

Artetxe et al. (2017) extend this approach to word2vec, successfully bootstrapping a cross-lingual word embedding space from a seed dictionary while preserving the performance of the learned cross-lingual word embeddings. They also advanced a step further and performed multiple iterations of bootstrapping, using the result of the previous iteration to induce a new seed dictionary. They showed that this iterative process, which they

dubbed *self-learning*, can significantly reduce the demand on the size of the initial seed dictionary.

The self-learning scenario was further extended to a fully unsupervised setting through an induction of the initial seed dictionary through unsupervised methods (Artetxe et al., 2018b). Their self-learning approach can be used with any mapping-based cross-lingual embedding construction method; however, there are no experiments to show this. Therefore, the motivation of this work is to investigate whether CCA-based mapping can be incorporated into the self-learning scenario.

Unsupervised mapping-based approaches have been proposed to relax the requirement of large dictionaries in supervised mapping based approaches (Artetxe et al., 2017, 2016). However, such efforts also rely on linear mapping-based methods, which raised my interest in incorporating KCCA into the unsupervised scenario. Therefore, I also investigate whether KCCA would improve the performance of existing linear-mapping-based unsupervised methods.

### 4.2.3 Self-Learning

As discussed in Section 3.4, the supervised cross-lingual word embedding approach first learns a mapping based on a dictionary, then uses the dictionary induction approach (See Section 3.6) to induce a new dictionary, which is then evaluated by a gold standard. This framework is formulated in Algorithm 1.

**Algorithm 1** The supervised cross-lingual word embedding

Input: Dictionary D, source word embeddings $\tilde{X}_s$, target word embeddings $\tilde{X}_t$
Step 1: $W$ = Learning mapping $(D, \tilde{X}_s, \tilde{X}_t)$
Step 2: $D_{new}$ = Learning dictionary $(\tilde{X}_s, \tilde{X}_t, W)$
Step 3: Evaluate($D_{new}$)

The self-learning scenario assumes that the training dictionary is used to learn a mapping, which induces a *larger and better* dictionary. Therefore, based on this assumption, the output dictionary is used as the input of the same system in a self-learning fashion. If the output dictionary is better than the previous one, it could hypothetically learn a better mapping and, consequently, an even better dictionary in the next iteration. Algorithm 2 summarises this idea.

As shown in Figure 30, the dictionary is first used to learn a mapping (e.g. CCA). The learned mapping is then maps the whole vocabulary of the two languages into a shared embedding space. Subsequently, the nearest neighbours approach (e.g. CSLS) is applied

**Algorithm 2** Self-learning: an unsupervised cross-lingual word embedding

Input: Source word embeddings $\tilde{X}_s$, target word embeddings $\tilde{X}_t$
Step 0: Seed dictionary initialisation (or given a weakly-supervised seed dictionary)
**repeat**
  Step 1: $W$ = Learning mapping $(D, \tilde{X}_s, \tilde{X}_t)$
  Step 2: $D_{new}$ = Learning dictionary $(\tilde{X}_s, \tilde{X}_t, W)$
**until** convergence
Step 3: Evaluate($D_{new}$)

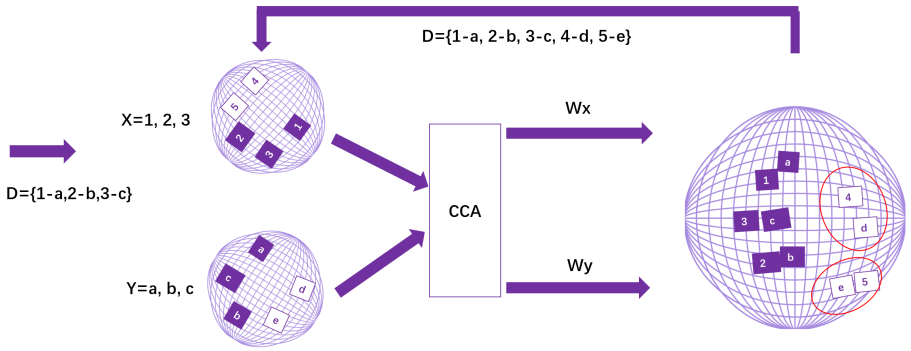



**Figure 30:** CCA-based self-learning scenario. The seed dictionary D is separated as two matrices of word embeddings x and y. CCA is applied on x and y seeks to find two projection matrices $w_x$ and $w_y$. Then the whole vocabulary is projected into the new feature space. Nearest neighbours are used to create a new dictionary and the whole operation is repeated.

to find the top K most similar word pairs from two languages to form a new dictionary. If the new dictionary is better than the original dictionary, the new dictionary can then be iteratively supplied into the system and deliver even better dictionaries at each step.

Based on this strategy, the model could run with a very small dictionary (even no dictionary) to learn a better, more extensive dictionary when the system converges.

## 4.3 Pipeline

In practice, the unsupervised cross-lingual word embedding approaches consist of four components:

1. The original seed dictionary — The seed dictionary is often given by some common concept of all languages like numbers (0-9). But this is not a must for the scenario.

2. Mapping-based approach. This part is further discussed in Section 3.

3. Dictionary induction approach.

4. Final step embedding mapping. Normally, after the divergence is finished, the final step, learning projection, is then applied to the dictionary finally learned.

### 4.3.1 Unsupervised Dictionary Initialisation

As illustrated in Algorithm 2, self-learning start learning with a weak supervision seed dictionary. Artetxe et al. (2017) use numerous dictionaries which consist of words matching the [0-9]+ regular expression in both vocabularies. Also, there are experiments involving randomly picked word pairs from the original training dictionary, which can roughly be used as a weakly-supervised dictionary.

The fully unsupervised dictionary initialisation approach assumes that the similarity matrices of source word embedding matrix $M_s$ and target word embedding matrix $M_t$ should be isometrically equivalent if the word embeddings are learned well. Also, the axis permutation order of $M_s$ and $M_t$ should be equivalent. Therefore, there is a possibility that, if the best matches of $M_s and M_t$ can be found, those indices then can be used to form a dictionary (at least a small dictionary in reality).

The model calculates similarity matrices with the word embedding matrices $\tilde{X}_s$ and $\tilde{X}_t$. The measure of similarity is the Euclidean distance:

$$Euc(x, y) = \|x - y\|^2 \tag{4.1}$$

Since the word embedding matrices are already length normalised, the Euclidean distance is directly related to cosine distance:

$$\cos(x, y) = 1 - \frac{\|x - y\|^2}{2} \tag{4.2}$$

Hence, the cosine distance and Euclidean distance could be seen as the same measure. For the similarity calculation, the cosine similarity is used to measure the distance between the word embeddings. Therefore, the similarity matrices can be formulated as:

$$M_s = \tilde{X}_s \tilde{X}_s^T \tag{4.3}$$

$$M_s = \tilde{X}_t \tilde{X}_t^T \tag{4.4}$$

Since the permutation combination is too large, it is computationally expensive to find the best matches. The value of each row is sorted, and obtain $sort(M_s)$ and $sort(M_t)$. In theory, words with the same meanings should have the same vector across languages (Ruder et al., 2019). Hence, given a word and its row in $sorted(M_s)$, its nearest neighbour over the rows of $sorted(M_t)$ is its corresponding translation.

Subsequently, the square root of the similarity matrix works better (Artetxe et al., 2018b). In practice, SVD is applied to the original word embedding matrices. For simplicity, I use source word embedding matrix $X_s$ as an example:

$$\tilde{X}_s = USV^T \tag{4.5}$$

The similarity matrix could be expressed as follows:

$$M_s = US^2U^T \tag{4.6}$$

The square root of $M_s$ is:

$$\sqrt{M_s} = USU^T \tag{4.7}$$

As described before, the top K nearest neighbours of $\sqrt{M_s}$ and $\sqrt{M_t}$ are selected as the word pairs of the seed dictionary.

### 4.3.2 Embedding Mapping

There are many mapping-based approaches reviewed in Section 3.3. The pipeline is flexible on all mapping-based methods. Artetxe et al. (2017) aims to map from source to target language, Artetxe et al. (2018b) used an SVD-based approach to map both source and target word embedding matrix into a shared embedding space.

### 4.3.3 Dictionary Induction

Dictionary induction means using the learned mapping to learn a new dictionary. Given two word embedding matrices $\tilde{X}_s$ and $\tilde{X}_t$, their mapped embeddings, denoted by $\tilde{X}_s^* = \tilde{X}_s W_s$ and $\tilde{X}_t^* = \tilde{X}_t W_t$, are used to find the nearest word pairs. These word pairs (denoted by $(x_s^d, x_t^d)$, $d < min(length(\tilde{X}_s), length(\tilde{X}_t))$) are obtained through some nearest neighbour approaches.

$$x_s^d, x_t^d = nn(\tilde{X}_s^*, \tilde{X}_t^*) \tag{4.8}$$

where $nn$ represents a nearest neighbours approach. Then, the indices of those word pairs in their corresponding vocabulary become the new dictionary, which is then used to find the next (and a hopefully an improved) mapping.

## 4.4 Proposed Investigation: Learning Unsupervised Cross-lingual Word Embedding with Non-linear Mapping

In summary, the previous works were boosting-like methods, aiming to find a better dictionary and a better mapping through iteratively linear mapping. This assumption is based on the fact that the word embeddings learned by word2vec can perfectly represent all of the words in languages; with the central concept being that the distance between languages should be isometric. However, this assumption does not hold for low-resource languages (e.g. Russian) or languages that are not in the same language family (e.g. Indo-European language family and Sino-Tibetan language family). The difference in grammar, concepts, and the writing system dramatically impacts the word orders in sentences, which may profoundly affect the word2vec representations. For example, Chinese, English and Spanish:

*Chinese*:
西汉所尊崇的儒家文化成为当时和日后的中原王朝以及东亚地区的社会主流文化。

*English:*
The Confucian culture respected by the Western Han Dynasty became the mainstream social culture of the Central Plains Dynasty and East Asia at that time and in the future.

*Spanish:*
La cultura confuciana respetada por la dinastía Han Occidental se convirtió en la cultura social principal de la dinastía de las Llanuras Centrales y el este de Asia en ese momento y en el futuro.

This example shows that in a similar language family, languages have similar grammar (English and Spanish are in the Indo-European language family), which suggest individual words with similar concepts have a higher probability of sharing similar contexts. However, words in a different language family (Chinese is in the Sino-Tibetan language family), words with similar concepts (denoted with the same colour), have a higher probability of having a different context. The diversity in grammar causes different word arrangements, which potentially further affects the word2vec representation. Previous linear-mapping-based boosting-like approaches may not work for the following two rea-

sons:

1. Linear mapping approaches are based on a concept that words with the same concept shall have the same geometric arrangement across-languages, but this assumption may not hold for many language pairs.

2. Iterative mapping assumes that, in every loop, a better mapping can lead to a better dictionary.

However, based on the first reason, the mapping of word embeddings from languages in different language families can possibly be non-linear.

Motivated by those reasons, I proposed to investigate whether non-linear mappings could learn a better dictionary in unsupervised self-learning settings. In addition, I proposed to further investigate whether different linear mapping approaches could affect the final dictionary quality.

The three key steps in the unsupervised mapping-based approaches are unsupervised dictionary initialisation, self-learning and the final projection. The initial goal was to investigate KCCA as the projection method in self-learning. However, I was unable to implement KCCA on tbe GPU, due to the very high memory requirements of the algorithm, while the CPU implementation was too slow to investigate the method in detail within the given hardware and time constraints. I opted to investigate the following approach instead. Since CCA can be run on the GPU efficiently, CCA was chosen as the projection method in the self-learning iterations (see Figure 30). Once CCA-based iterative mapping step converged, the KCCA was applied as the final projection step using the dictionary induced by the last step of CCA, to produce the new shared representation. The overall process is summarised below.

### 4.4.1 Step One: Unsupervised Dictionary Initialisation:

The unsupervised dictionary initialisation described in 4.3.1 is employed.

### 4.4.2 Step Two: Self-learning

The first iteration of self-learning uses the seed dictionary produced in Step one. The other steps use the dictionary that is induced in the previous step. In each iteration, the model learns a mapping using the dictionary in the Step one; Step two maps the two vocabulary embedding matrices into the shared embedding space; The nearest neighbours approach

is then used to induce a new dictionary (See Section 4.3.3. The dictionary is then used in step one of the next iteration. The proposed model uses the CSLS (See Equation 3.55) to measure the similarity.

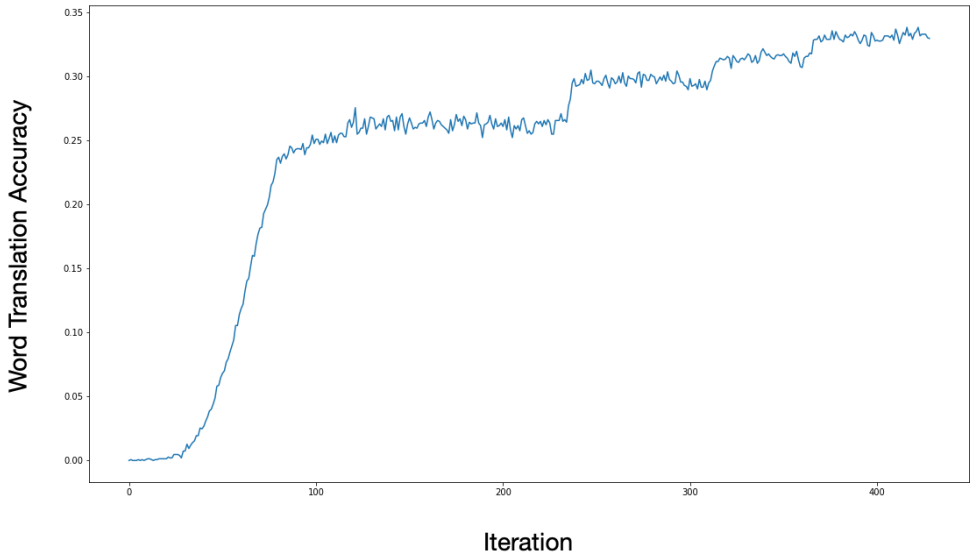
In practice, there are also some additional tricks which can further improve the dictionary induction process:

- **Iteratively CCA-based mapping.** The self-learning is adapted to SVD-based approaches, but not CCA. Therefore, one motivation of this research is to investigate whether CCA can achieve comparable or even better results in a self-learning scenario

- **Stochastic Dictionary Induction.** The mapping-based approaches tend to easily fall into local optima. Therefore, I adapt an element drop-out trick described in Artetxe et al. (2018b): During retrieval in every iteration, part of the element in the similarity matrices $M_s$ and $M_t$ are randomly selected, and those elements remain unchanged. Then the left elements are changed to 0. A probability $p$ is set to describe how many elements are changed or kept unchanged. As the iteration grows, $p$ is increased starting from 0.1, and the value is doubled when the objective function stops increasing for $n$ loops. A mask is defined and initialised as an identity matrix $m = I$. In an iteration, each component of $M$ will change to 0 with a probability $p$, and the new mask $m^*$is obtained. Then given the similarity matrix $M$ between $\tilde{X}_s$ and $\tilde{X}_t$, a masked similarity matrix is then used in the following dictionary induction process, $M = M * m$. The probability $p$ changes to (0.1, 0.2, 0.4, 0.8, 1) when a convergence criterion is reached. Figure 31 illustrate the stochastic dictionary induction process. In the start, the quality of the learned dictionary gradually improves but it tends to fall in local optima soon. After $p$ is set gradually larger, the proposed model jumps out of the local optima, and the model has a another chance to find a better optimum.

- **Bi-Directional Dictionary Induction.** In practice, the single direction (from source to target) dictionary induction process tends to ignore words in target languages, which may cause the model to quickly fall into local optima during iterations. Therefore, the bi-lingual dictionary induction process is proposed. As described in Equation 3.55, the top 1 translation from source to target is selected and then appended to a new dictionary $D_{s\rightarrow t}$. The dictionary $D_{t\rightarrow s}$ from target to source is

**Figure 31:** Word translation accuracy of unsupervised CCA-Based cross-lingual word embedding on the test set of En-Es dataset.




obtained using the same approach. Finally, concatenating $D_{t \to s}$ and $D_{s \to t}$ and a new dictionary D is obtained as the learned dictionary, which is then used to learn mapping for the next iteration.

- **Vocabulary Cutoff.** The similarity matrix between $X_s$ and $X_t$ is large, so it is computationally expensive to calculate it in each loop. In addition, some low-frequency words are not ideal for mapping. Therefore, I only use top K words from the source and target words in terms of word frequency to learn new mappings in each loop iteratively.

### 4.4.3 Step Three: Final Projection

KCCA, as described in Section 3.5, is applied to learn the final shared embedding space. It uses the output dictionary of the final self-learning step. However, this dictionary can be fairly large due to the significant size of the vocabularies. Since the size of the kernel matrix is the same as the number of elements in the dictionary squared, it is impractical to use the whole dictionary. I propose using the mutual nearest neighbour method for producing this final dictionary which reduces dictionary entries to a more manageable number, making KCCA faster and reducing memory requirements. In the experiments, it is also found that the mutual nearest neighbour criterion enhances the quality of the dictionary, making KCCA more effective. The proposed work is summarised in Algorithm 3.

**Algorithm 3** Proposed: KCCA-based unsupervised cross-lingual word embedding

Input: Source word embeddings $X_S$, target word embeddings $X_T$
Step 0: Seed dictionary initialisation (or given a weakly-supervised seed dictionary)
**repeat**
  Step 1: $W$ = CCA $(D, \tilde{X}_s, \tilde{X}_t)$
  Step 2: $D_{new}$ = Dictionary Induction $(\tilde{X}_s, \tilde{X}_t, W)$
**until** convergence
Step 3: $D_{final} = KCCA(D_{new}, \tilde{X}_s, \tilde{X}_t)$
Step 4: Evaluate($D_{final}$)

### 4.4.4 Convergence Criterion

The convergence criterion, also described as the global objective, is used to describe the distance of the cross-lingual word embeddings from different languages. Given two dictionary embedding matrices $X_s$ and $X_t$, and learned mapping $W_s$ and $W_t$, the learned cross-lingual word embedding of the words in the dictionary can be obtained using Equation 4.9 and 4.10:

$$X_s^* = X_s W_s \tag{4.9}$$

$$X_t^* = X_t W_t \tag{4.10}$$

Then the convergence criterion is defined as the sum of its similarity matrix. For computational simplicity, the cosine similarity is used instead of CSLS for measuring distance in the convergence criterion:

$$J = \sum X_s^* X_t^{*T} \tag{4.11}$$

When J stops increasing for several iterations (I set a patience parameter q), or J is smaller than a number $\epsilon$, the system is deemed to have stopped improving and the proposed model ceases updating.

## 4.5 Experiments

### 4.5.1 Dataset

The dataset described in Section 3.6 is used, which includes two parts:

1. The English-Italian (En-It) dataset proposed by Dinu and Baroni (2015), which includes the 300-dimensional pre-trained monolingual word embeddings of English and Italian, a training dictionary and a test dictionary.

2. The English-German (En-De), English-Spanish (En-Es), English-Finnish (En-Fi) dataset proposed by Artetxe et al. (2017). This dataset is an extension of Dinu and Baroni (2015), with the same word2vec training process and training & testing dictionary selection process.

However, since the proposed approach is unsupervised, only the monolingual corpora are available. This means the training dictionaries of all datasets mentioned are not accessible to the proposed model. The test dictionary is only used for the evaluation.

### 4.5.2 Experiment Details

The experiment investigates the benefits of CCA in the fully unsupervised scenario. First, CCA is implemented as the word embedding mapping method in each iteration of self-learning. It is evaluated against the best existing mapping approach - SVD - that is applied to the same unsupervised seed dictionary. The evaluation uses the same task and test set as the first experiment. Second, KCCA is investigated as the final step projection method and compared to the last step projection method used by Artetxe et al. (2018a). This entailed, Whitening, re-weighting, SVD-based projection, de-Whitening and dimensionality reduction (or simply multi-step mapping), which has been shown to improve accuracy by about 1 percentage point. For completeness, KCCA is also applied as the final step projection to both CCA- and SVD- based self-learning to investigate whether there is an advantage in using both methods.

In stochastic dictionary induction, $\epsilon = 0.006$. An patience parameter $q = 10$, when the performance of the model stop growing for $q$ loops, the model is considered to be convergence and stop updating. The vocabulary cutoff parameter $n = 20000$ for computational simplicity.

The implementation uses NumPy[1]. Experiments for the self-learning part make use of CuPy[2] to accelerate the algorithm using CUDA. All experiments were carried out on AMD Threadripper 1950x 16-core CPU with 128Gb RAM. CUDA-accelerated code was executed on Nvidia RTX 2080 Ti GPU using CUDA 10.

### 4.5.3 Evaluation

The word translation accuracy (also known as precision @1) is used to measure the performance of the final learned dictionary.

[1] https://numpy.org/
[2] https://cupy.dev/

## 4.6 Result Analysis

### 4.6.1 CCA-based Self-learning Iterations

Figure 32 shows that the SVD-mapping-based self-learning model has longer training iterations than the CCA-mapping-based model. CCA achieves a better local optimum and has a shorter training iteration. In En-De dataset, the CCA-mapping-based system has more training iterations but finds better local optima.

Table 4.2 shows the results of using CCA- based mapping in self-learning iterations. The model is evaluated straightly after self-learning iterations without any additional projection steps. The result is compared to Artetxe et al. (2018a). The results show that CCA outperforms SVD by 3.0 and 2.7 percentage points for English- German and English-Spanish language pairs; however, it under-performs by 0.6 and 0.76 points for English-Italian and English-Finnish language pairs.

The results confirm that CCA can be successfully used in the self-learning framework. Although it cannot be conclusively claimed that it is better than the traditional SVD approach for all languages, the results indicate that it is advantageous for some language pairs. It is an approach that should be considered especially considering that a generalised version of CCA can produce a shared embedding for more than two languages.

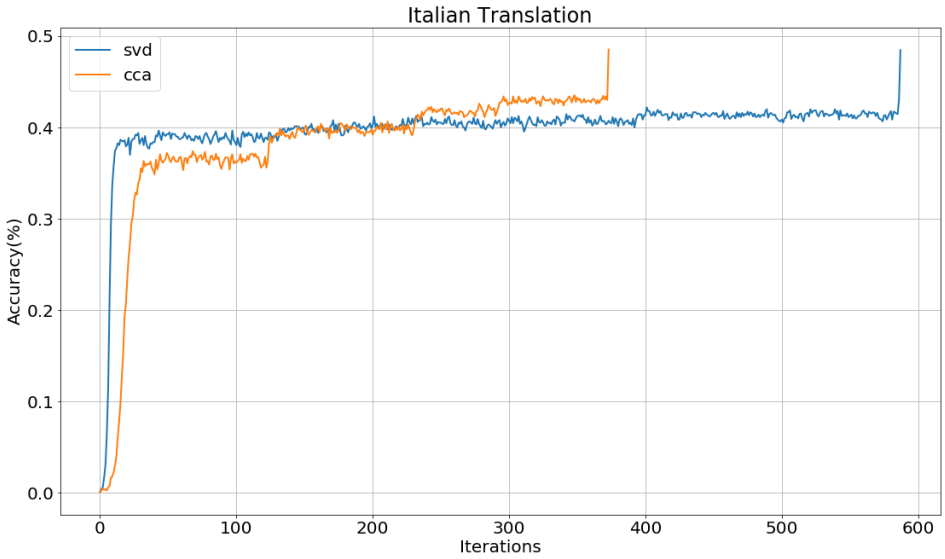



**Figure 32:** Word translation accuracy of test set during iterations on En-It dataset. Using CCA-based mapping in self-learning process. The final step mapping is KCCA mapping.

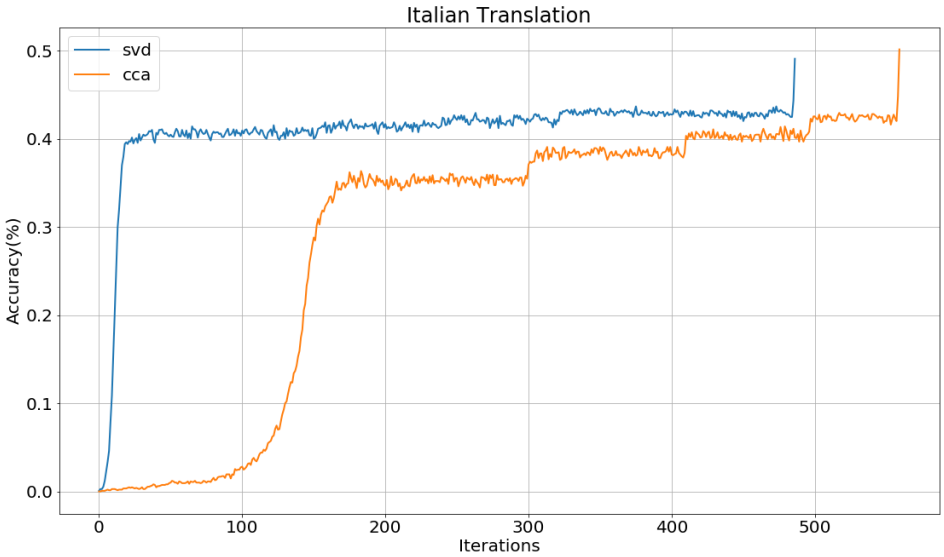



**Figure 33:** Word translation accuracy of test set during iterations on En-De dataset. Using CCA-based mapping in self-learning process. The final step mapping is KCCA mapping.

### 4.6.2 Final Projection Step

The proposed approach is compared with an orthogonal-mapping based final step projection proposed by Artetxe et al. (2018b), who adapted the Whitening, reweighing and de-Whitening process in the final projection process. Figure 35 and Figure 34 show the reproduction of this final mapping. Figure 32 and Figure 33 show the proposed KCCA-based final projection step.

Table 4.3 shows the detailed results of applying either KCCA or orthogonal multi-step mapping or both as the final projection step after CCA-based self-learning. The application of KCCA alone produces the largest improvement over the self-learning baseline of 1.3, 2.0, 3.1 and 2 percentage points for English-Italian, English-German, English-Spanish and English-Finnish language pairs respectively in terms of word translation accuracy. At the same time, multi-step mapping produces almost no improvement or even degrades the accuracy. Table 4.4 shows the results of applying either KCCA or multi-step mapping as the final projection step after traditional SVD-based self-learning. The application of KCCA alone improves the accuracy more than the conventional multi-step mapping alone for all language pairs. However, for English-Italian and English-Spanish, applying them together improves results even further, although only slightly. Figure 36 and Figure 37 illustrate the final step's improvement. The non-linear-based final step projection has a significant performance jump for all language pairs.

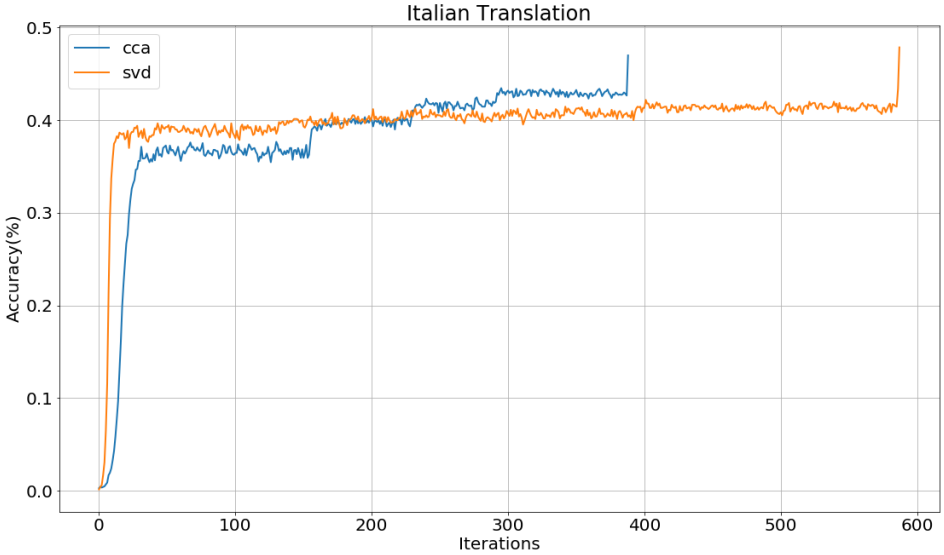



**Figure 34:** Word translation accuracy of test set during iterations on En-It dataset. The CCA-based mapping approach is used in self-learning process. The final step mapping is orthogonal mapping.

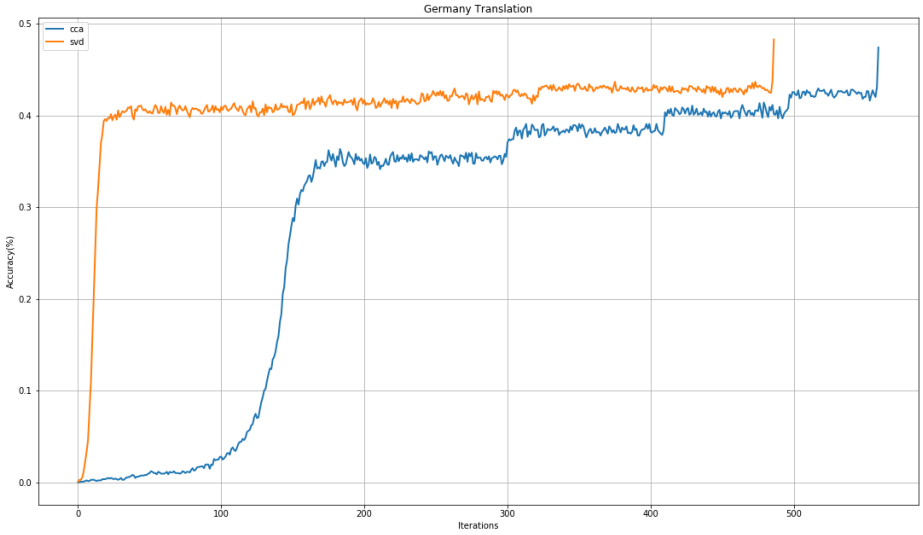



**Figure 35:** Word translation accuracy of test set during iterations on En-De dataset. The CCA-based mapping approach is used in self-learning process. The final step mapping is orthogonal mapping.

It is also worth noting that the experiment on the En-Fi dataset required more time than other datasets on finding the best matches. The proposed model takes about 50, 80, 200 iterations to find the initial reasonable dictionary (refer to the first stage) of En-Es, En-De, En-It dataset. However, the proposed approach takes nearly 2000 iterations to find this feasible initial dictionary. This is because the Finnish corpus is inadequate, making the

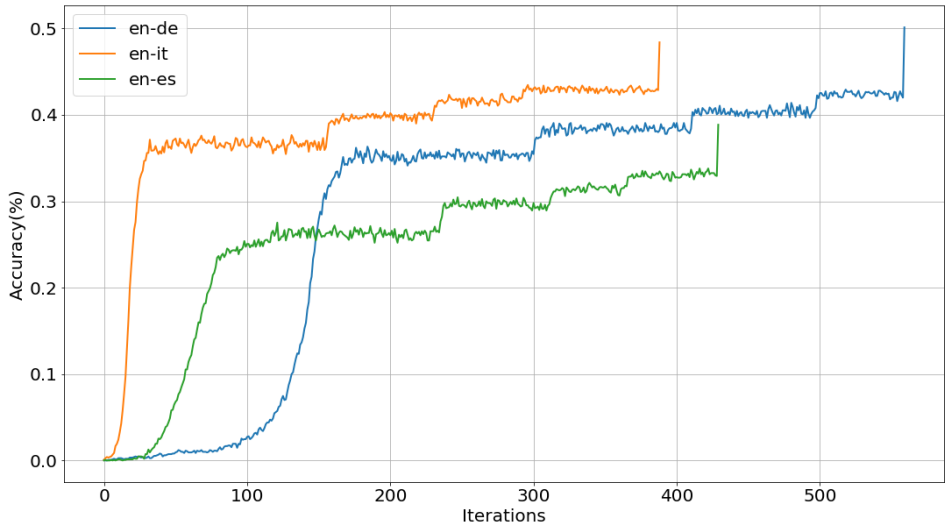



**Figure 36:** Word translation accuracy of test set during iterations on En-De, En-Es, En-It dataset.

monolingual word embeddings of Finnish perform worse than that of English, Spanish, French or Italian. Therefore, the quality of the monolingual word embeddings highly impacts the unsupervised methods. The data scarcity issue can lead to a sub-optimal result in the unsupervised models.

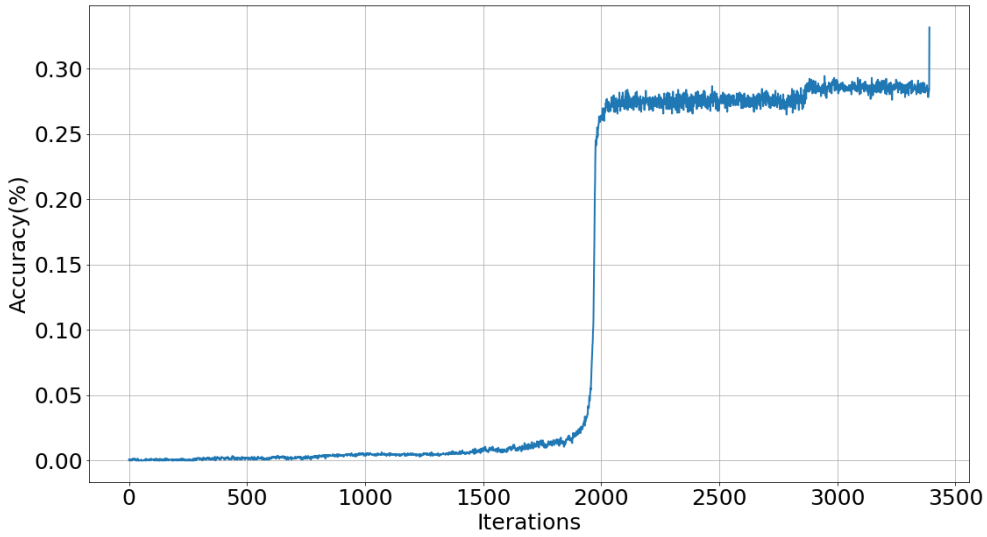



**Figure 37:** Word translation accuracy of test set on En-Fi dataset. The proposed system spend much more time on finding the first reasonable local optima.

**Table 4.2:** Accuracy(%) comparison of the different mapping based approaches in self-learning process.

| CCA | SVD | En-It | En-De | En-Es | En-Fi |
|---|---|---|---|---|---|
| ✓ | | 47.20 | **48.10** | **36.4** | 31.40 |
| | ✓ | **47.80** | 45.10 | 33.7 | **32.16** |

**Table 4.3:** Accuracy(%) analysis on the final step projection in CCA-based self-learning process.

| Typical Mapping | KCCA | En-It | En-De | En-Es | En-Fi |
|---|---|---|---|---|---|
| | | 47.20 | 48.10 | 36.4 | 31.40 |
| ✓ | | 47.00 | 47.40 | 37.0 | 31.60 |
| | ✓ | **48.53** | **50.13** | **39.47** | **33.50** |
| ✓ | ✓ | 48.40 | 49.86 | 38.93 | 32.93 |

**Table 4.4:** Accuracy(%) analysis on the final step projection in SVD-based self-learning process.

| Typical Mapping | KCCA | En-It | En-De | En-Es | En-Fi |
|---|---|---|---|---|---|
| | | 45.10 | 47.80 | 33.7 | 32.16 |
| ✓ | | 47.80 | 48.26 | 36.5 | 32.30 |
| | ✓ | 48.46 | **49.06** | 37.46 | **32.79** |
| ✓ | ✓ | **48.93** | 48.80 | **37.93** | **32.79** |

### 4.6.3 Comparison with State-of-the-art

Table 4.5 shows a comparison between the proposed approach and prior unsupervised or weakly supervised methods. The works conducted by Artetxe et al. (2018b) and Lample et al. (2018) are fully supervised cross-lingual word embedding approaches. The proposed model obtains the best result, offering a translation accuracy increase of 0.4, 1.7, 1.9 and 0.0 percentage points on English-Italian, English-German, English-Spanish and English-Finnish language pairs and achieves the state-of-the-art amongst existing approaches.

As shown in Table 4.5, I provide the best result reported in previous works.The work conducted by Artetxe et al. (2017) has two scenarios: The [25] means they used a manually selected dictionary of size 25; The [num] means they selected some number, which appears in all language pairs. Those numbers are chosen from different corpora and formulated as the seed dictionary.

## 4.7 Conclusion & Future Works

This chapter discusses learning cross-lingual word embeddings in an unsupervised scenario. I demonstrate that CCA can be successfully used as the projection method in the

**Table 4.5:** Accuracy(%) comparison of the previous unsupervised methods.

| Method | **En-It** | **En-De** | **En-Es** | **En-Fi** |
|---|---|---|---|---|
| Artetxe et al. (2017) [num] | 37.27 | 39.60 | - | 28.16 |
| Artetxe et al. (2017) [25] | 39.40 | 40.27 | - | 26.47 |
| Lample et al. (2018) | 45.40 | 47.27 | 36.20 | 1.62 |
| Artetxe et al. (2018b) | 48.53 | 48.47 | 37.60 | 33.50 |
| Proposed | **48.93** | **50.13** | **39.47** | **33.50** |

self-learning scenario and that it performs approximately on par with the current state-of-art SVD-based mapping. I also show that existing SVD-based self-learning methods could benefit from KCCA as the final projection step, either alone or combined with the existing multi-step mappings.

In the future, I am planning to extend the proposed work to other language pairs that may benefit even more from non-linear mappings, such as Chinese and Japanese. Moreover, I plan to investigate multi-lingual word embeddings by applying generalised KCCA.

CHAPTER 5

# Consistency-based Cross-Lingual Word Embeddings

The training dictionary plays an essential role in learning mapping-based cross-lingual word embedding methods (this is referred to as mapping-based methods from now on). The proposed investigation illustrates that a more extensive dictionary leads to better cross-lingual word embeddings. However, an extensive dictionary is often hard to obtain (Artetxe et al., 2018b; Lample et al., 2018; Artetxe et al., 2017). This chapter describes a semi-supervised method that can bootstrap a small dictionary into a more extensive and high-quality dictionary.

A crucial challenge that underlies the process of constructing an accurately augmented dictionary, is found in the *high variance* issue. An ensemble-like approach referred to as the *word consistency-based model* is proposed to address the challenge. The ensemble model attempts to control the high variance by generating multiple weak estimators and thereby introducing diversity into each estimator. The estimators are used to learn an augmented dictionary separately. Then a robust estimator is subsequently ensembled through a voting procedure termed as the *model agreement* from the weak estimators. The robust estimator built through the ensembling process is then used to learn a new dictionary. Extensive experiments using the ensembling approach are presented on a popular cross-lingual word embedding dataset, and the results illustrate the efficacy of the proposed method in comparison to existing state-of-the-art approaches (Lubin et al., 2019; Doval et al., 2018; Artetxe et al., 2017; Dinu and Baroni, 2015).

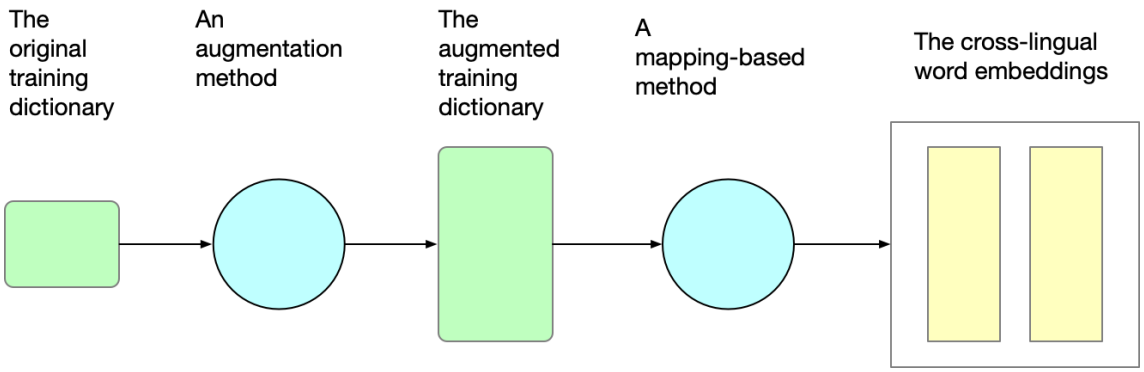



**Figure 38:** A brief summary of the dictionary augmentation models.

## 5.1 Introduction

### 5.1.1 Bias-Variance Tradeoff in the Mapping-based Methods

The mapping-based methods often suffer from the data insufficiency problem. This issue arises for two reasons as follows.

- **Inadequate training data for the monolingual word embeddings**. Low-resource languages (e.g., Finnish or Turkish) often lack monolingual training corpora, which further degrades the quality of the monolingual word embeddings. Monolingual word embeddings of low-resource languages also contain noise, so they are imprecise at describing the feature of words.

- **Insufficient training word pairs**. Low-resource language pairs (e.g., Chinese-Russian) are often language pairs with scarce parallel corpora. This issue leads to limited training dictionaries for mapping-based methods.

Both of the above issues lead to inadequate and low-quality dictionaries for effective use of mapping-based methods. Data augmentation is one option to mitigate these challenges. Several data augmentation methods have been proposed in recent years to address this. Artetxe et al. (2017) use a bootstrap-like method to generate a larger dictionary utilising a small seed dictionary. This method provides competitive results compared to the typical methods using manually generated dictionaries. Artetxe et al. (2018b) further extend this method which is fully exempted from the requirement of paired signals.

As illustrated in Figure 38, the above-mentioned approaches have a similar paradigm that generates a more extensive dictionary as a first step, and then uses the generated dictionary to map the monolingual word embeddings into a shared vector space. However, those methods only focus on the generated dictionary's quantity, i.e. they ignore

the quality of the dictionary. The augmented dictionary can only provide a very close (or slightly better) result than the manually generated training dictionary. Based on my research findings, I hypothesize that this is caused by poor handling of the variance-bias tradeoff. The noisy original training dictionary leads to *high variance, high bias* models. The augmented dictionary indeed has a larger size, proving to help deal with high bias issues (Yang et al., 2020b); however, the poor quality of the training dictionary still leads to a high variance issue.

### 5.1.2 The Ensemble Model

The intuitive scenario of dealing with the high variance issue is to use an ensemble (D'Ascoli et al., 2020). Briefly speaking, an ensemble-based approach relies on generating many separate models (weak estimators) using different methods like variations in the training datasets which enforce the necessary diversity of models; then, the outputs of those separate models are aggregated to produce a final result (Kuwabara et al., 2020; Hansen and Salamon, 1990).

The ensemble-based methods are able to alleviate the high variance issue because:

- **The averaging of different models can mitigate poorly performing models.** Due to the large amount of noise that may exist in the original training data, there is a high possibility that some of the training word pairs may have a disproportionately large negative impact on the final mapping step.
- **The ensemble model can effectively deal with high-dimensional data.** Simple mapping has difficulty in handling high dimensional training word embeddings.

Figure 39 illustrates an intuitive ensemble model for the cross-lingual word embeddings.

In summary, the key goal of this chapter is to develop techniques that address the bias-variance tradeoff of the mapping-based methods. The crux of this strategy centres on leveraging an ensemble-like model.The main contributions of this chapter are listed below:

- This chapter provides one of the first investigations into dealing with the trade-off between variance and bias in mapping-based cross-lingual word embedding models.

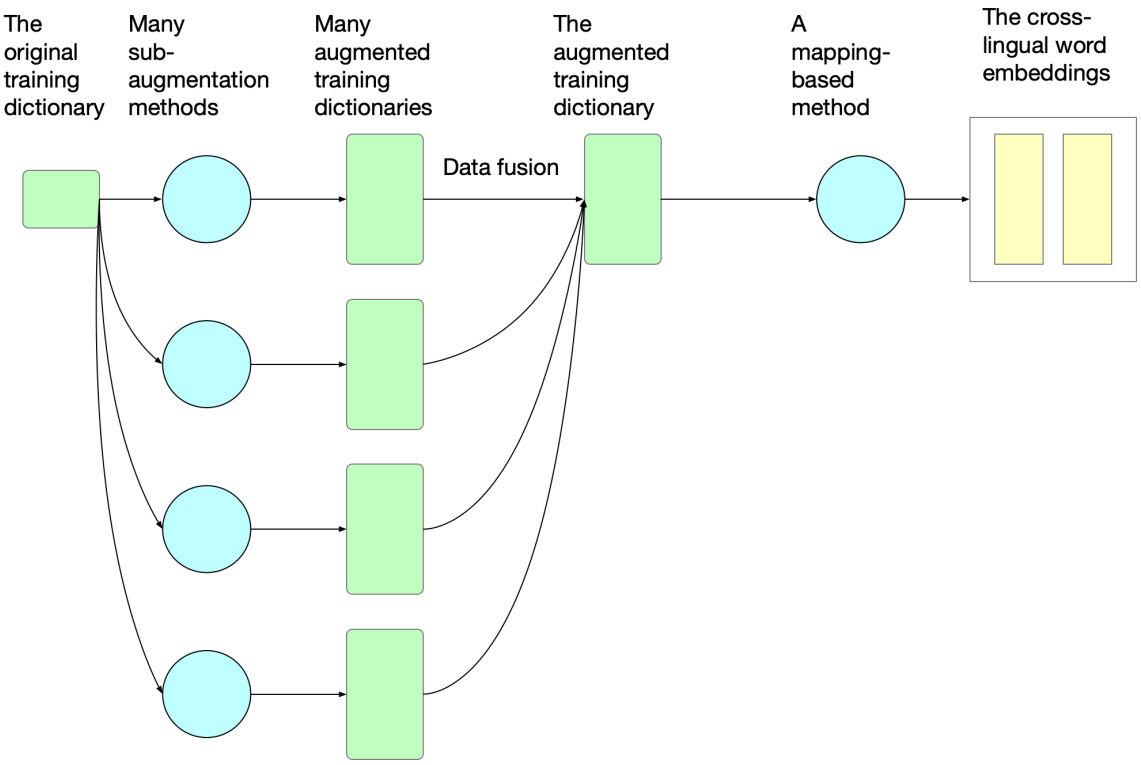


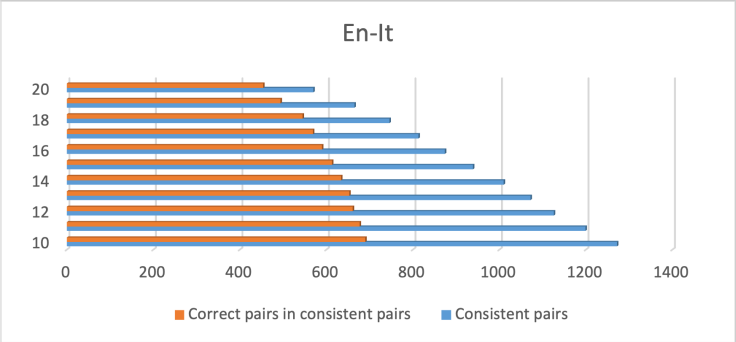

**Figure 39:** An ensemble-like dictionary augmentation model.

- I provide a robust and effective way to augment training data. The augmented high-quality dictionary can be used in mapping-based methods, which is vital for many low-resource languages.
- The performance of the learned cross-lingual word embeddings achieves the state-of-art among the mapping-based methods.

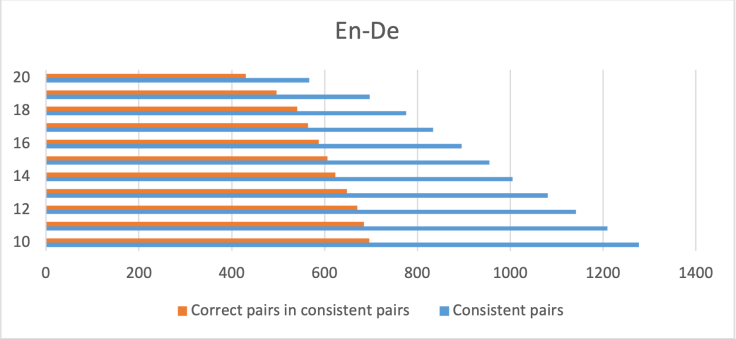
## 5.2 Bagging

Bootstrap aggregating (refer to as bagging in short) (Breiman, 1996) describes the foundations of my initial experiments with ensemble-based approaches. The bagging model works by randomly picking different subsets from training data with replacement. The different training subsets are then used to learn different variations of mapping. Since the training subset is a fraction of the superset, the learned mappings can be seen as 'weak' mappings. Once the ensemble has been created, a fusing method is used to find a final aggregated mapping. In this section, I focus on the first step and illustrate the result in Figure 40.

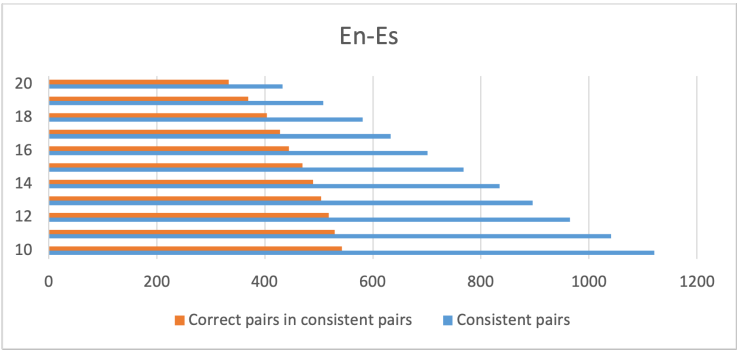
The investigation into the learned mappings has revealed an interesting phenomenon that points to some words tending to be translated into the same word in another language regardless of the underlying training set or the mapping methods used. I refer to those



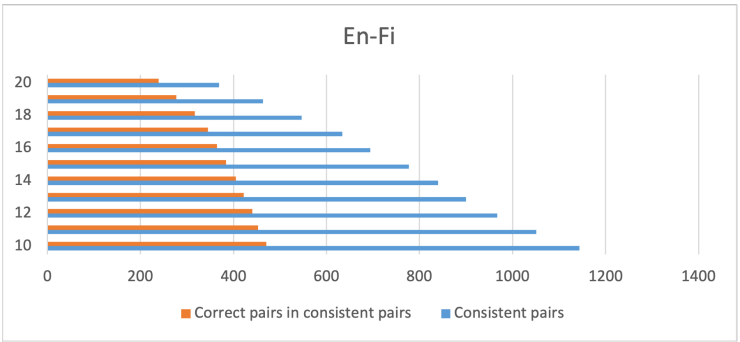

**(a)** En-It dataset.




**(b)** En-De dataset.




**(c)** En-Es dataset.




**(d)** En-Fi dataset.

**Figure 40:** Consistent word pairs of test sets in a different number of models. The y-axes represent how many models are used. The x-axes represent the number of word pairs. The blue bars are the consistent pairs and the orange bars represent the correctly translated word pairs that are in the consistent pairs.

words as *consistent* words. As shown in Figure 40, the word translation accuracy, or the number of correctly translated word pairs, is higher in the group of consistent words. I hypothesize that the consistent words, which can accurately be extracted from two monolingual corpora, can be used to augment the training dictionary *reliably*.

The experiments are conducted on four language translation datasets. English-Finnish, English-German, English-Spanish and English-Italian. The experimental design consists of re-initializing the training dictionary for each language pair by sampling it $n$ times with replacement. This sampled dictionary is used as the new dictionary and is applied to the cross-lingual word embedding pipeline. The resulting pipeline, described in Section **3.3.2**, is implemented using the same settings described in Artetxe et al. (2018a).

I sample new dictionaries from the original training dictionary with replacement and generate {10, 12, 14, 16, 18, 20} sub-training sets. A "weak" mapping is then learned from those training subsets. Subsequently, two types of result outputs are collected for testing as follows.

- The number of English words translated to the identical target translation by multiple ensembles. Those word pairs are defined as consistent pairs, denoted as blue lines in Figure 40.
- The correctly translated word pairs within the set of consistent pairs, denoted by orange lines. Results from the four language-pair translations are illustrated in Figure 40.

A deeper analysis into the translation quality of cross-lingual word embeddings uncovered a noteworthy pattern. It was observed that numerous words tend to be translated into the same results despite the differences in the training dictionary. Additionally, it can be seen from Figure 40 that as the ensemble size grows, two properties emerge: The number of consistent pairs decreases; The proportion of the correctly translated pairs in consistent pairs increases. I define this phenomenon as *consistency*.

This reveals two attributes regarding these types of words: First, they have strongly correlated nearest neighbours; Second, they share similar geometric arrangements. Based on these two attributes, I hypothesise that, as the frequency of a word translated by different weak mappings to the same translation increases, the probability, in turn, increases that this consistent word pair is the ground truth, which can then be appended to the original dictionary.

This hypothesis provides a mechanism for constructing language-pair dictionaries in a semi-supervised manner. **As more consistent pairs are generated within all cross-lingual word embeddings in a language pair, it turns out that those language pairs can be used to construct a larger and a higher-quality dictionary.** This approach is termed as *Consistency-based cross-lingual word embeddings*.

Intuitively, the word consistency has two forms, cross-consistency and self-consistency. Cross-consistency indicates the consistency between different mapping-based methods (KCCA-SVD, CCA-SVD). Self-consistency reflects the word consistency in an identical model with varying samples of training (KCCA-KCCA, SVD-SVD).

In order to develop this concept, the translation result of different models are analysed. The translation result of different models are separated into six groups listed below.

- **Consistent and correct**. The number of words that are correctly translated by both methods. This means the words have very strong relationships (A very high similarity) in a different feature space.
- **Consistent but incorrect**. The number of words that are incorrectly translated by both methods. This means the words have a very strong relationships (A very high similarity) in a different feature space.
- **Inconsistent-1-correct & Inconsistent-2-correct**. The word pairs that are only correctly translated by a specific model (e.g., only correctly translated in KCCA-based method but incorrectly translated by CCA)
- **Inconsistent-1-incorrect & Inconsistent-2-incorrect**. The word pairs that are only incorrectly translated by a specific model (e.g., only incorrectly translated in KCCA-based method but correctly translated by CCA).

The possible outcome categories are a helpful lens for conducting a deeper analysis of mapping-based methods. This is especially relevant for consistent word pairs because if it can be determined what kinds of words tend to have a higher similarity among all languages, then potentially a better dictionary can be generated.

### 5.2.1 Cross-Consistency

The word cross-consistency approach is a practical analytical approach for evaluating the performance of different methods. I am interested in knowing what kind of words turn out to be correctly translated by various mapping approaches. Suppose KCCA can

correctly translate a word pair. In that case, it can be assumed that this word pair has a high possibility of sharing a non-linear relationship and vice versa. If both linear and non-linear methods can correctly translate a word pair, it is reasonable to believe this word pair may share a very high geometric similarity.

Figure 41a and 41b shows the distribution of correctly and incorrectly translated words among consistent and inconsistent word pairs in the En-Zh dataset. The correctly translated word pairs occupy a considerable proportion among all consistent word pairs, and the incorrectly translated words occupy a larger proportion among all inconsistent word pairs. Figure 41c and 41d also reveal a similar trend that many words in the test set can be mapped correctly by both linear and non-linear methods. I hypothesize that this kind of word pairs has a stable relationship that can be easily aligned regardless of linear or non-linear mapping.

Additionally, incorrectly translated words (red bar) appear resistant to a correct translation by linear or non-linear methods. Two possible factors may cause this phenomenon: Those word pairs are noise, so the test set contains outliers and needs improvement; The evaluation method is not robust and also needs to be modified.

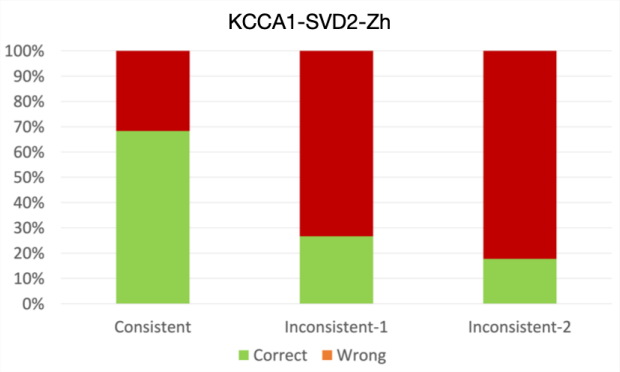
Figure 41a and 41c also reveals that there are more inconsistent pairs in the En-Zh test set than in the En-Es dataset. This also further supports the assumption in Section 3.3 that non-linear methods perform better in the En-Zh dataset.

### 5.2.2 Self-Consistency

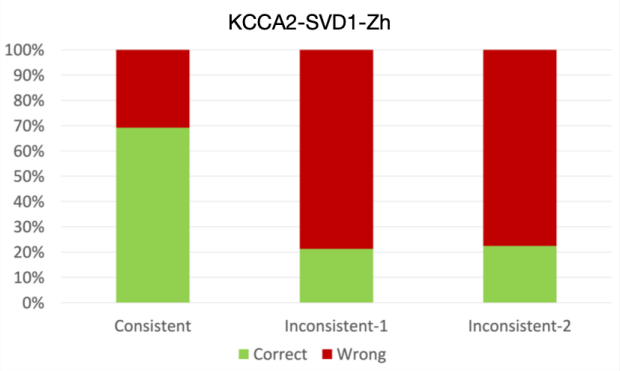
Based on the ensemble theory, the different training subsets can provide diversity in each weak model and improve the robustness of the final ensemble model. Self-consistency offers insight into the robustness of an identical method by training with different data subsets.

Figure 42 and 43 shows that separate sub-models can yield similar self-consistency results in different datasets. The correctly translated word pairs among consistent word pairs are proportionally larger than those of inconsistent ones in both datasets. This property is of interest as it may indicate that, **if the same method can provide different results with different training inputs, then it could be possible to ensemble those models (same method, different training subset) to learn a better mapping, as well as a better dictionary.**

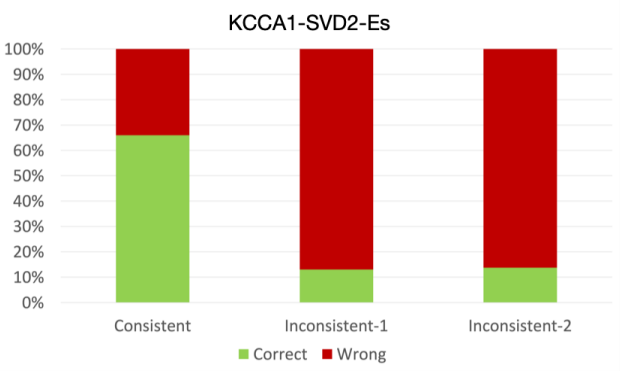
Subsequently, Figure 42 and 43 illustrate that the inconsistent but correctly translated pairs (green bar of inconsistent-1 or 2-wrong) occupy a sizable proportion of the total



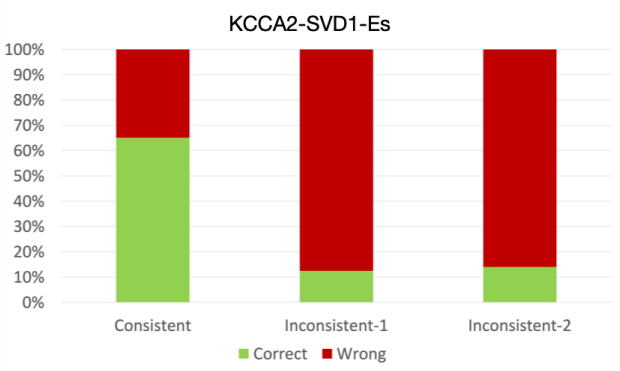

**(a)** Cross-consistency between KCCA- and SVD- based methods on En-Zh dataset. Parameter set one.




**(b)** Cross-consistency between KCCA- and SVD- based methods on En-Zh dataset. Parameter set two.




**(c)** Cross-consistency between KCCA- and SVD- based methods on En-Es dataset. Parameter set one.




**(d)** Cross-consistency between KCCA- and SVD- based methods on En-Es dataset. Parameter set two.

**Figure 41:** The cross-consistency on En-Zh and En-Es dataset. The figures show the result of two sets of parameters of KCCA- and SVD- based mapping.

translation results. Those words are often treated as noise in the previous works. I hypothesise that if a model has a low self-consistency, noise can be alleviated by an ensemble with the same model trained by different training sets. Therefore, I posit a self-consistency-based ensemble model which aims to generate a better dictionary.

## 5.3 Consistency-based Cross-Lingual Word Embeddings

Based on the hypothesis mentioned above, a consistency-based model is designed, which is used to generate a large and high quality dictionary. As shown in Figure 44, the proposed approach can be formulated into four steps.

1. **Sub-training set sampling**. $n$ translation word pairs from the original training set are sampled for each weak model, thus obtaining $I$ training sets.

2. **Linear mapping**. For each weak model, a linear mapping is applied to the monolingual word embeddings and from this, the cross-lingual word embeddings are obtained.

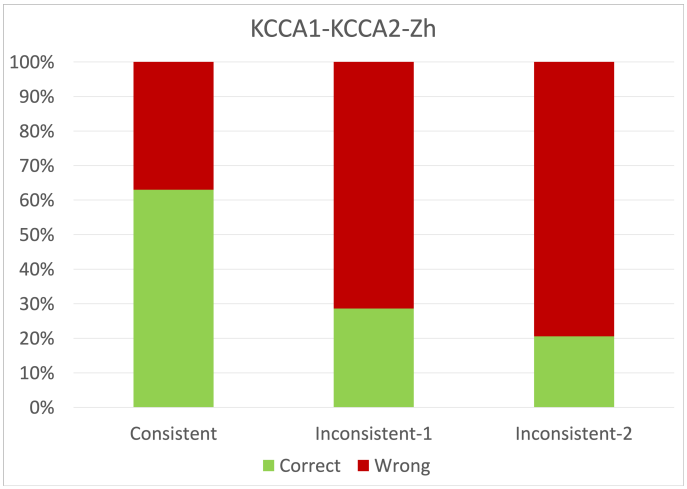
3. **Model agreement**. In each generated weak model, the word pairs that are nearest neighbours are selected using a dictionary induction method. In order to do this, a similarity matrix $M$ is calculated:

$$M = O(\tilde{X}, \tilde{Y}) \tag{5.1}$$

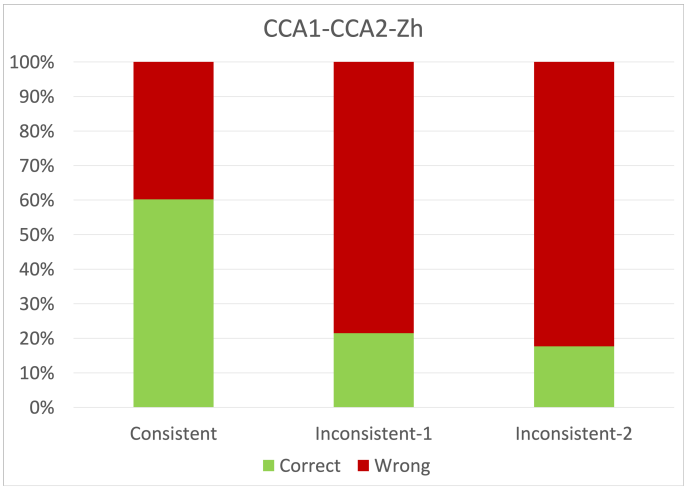
   Next, word pairs that appear in all $I$ weak models are collected. This process is defined as the *model agreement*, signifying that the selected word pairs are consistent word pairs across all weak models. Subsequently, if two different dictionaries produced by different models have the same word pair, those models $agree$ this word pair is consistent. If a word pair is $agreed$ by all $I$ models, the final model considers this word pair an accurate translation. Figure 45 depicts this process.

4. **Combining**. Finally, the consistent word pairs obtained in step four and the original training dictionary are concatenated to form a larger new dictionary.

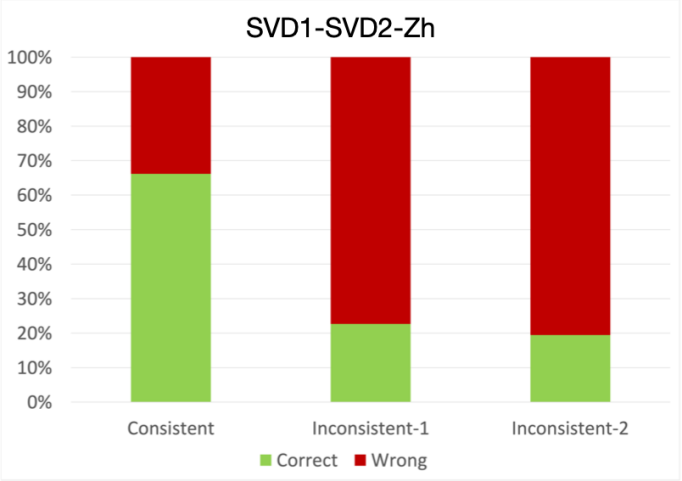
Algorithm 4 outlines the proposed method and Table 5.2 exhibits the experimental results. Figure 40 demonstrates that if more models agree on a word pair, this word pair tends to be accurate. This result supports the stated hypothesis. Also, Table 5.2 further




**(a)** KCCA's self-consistency

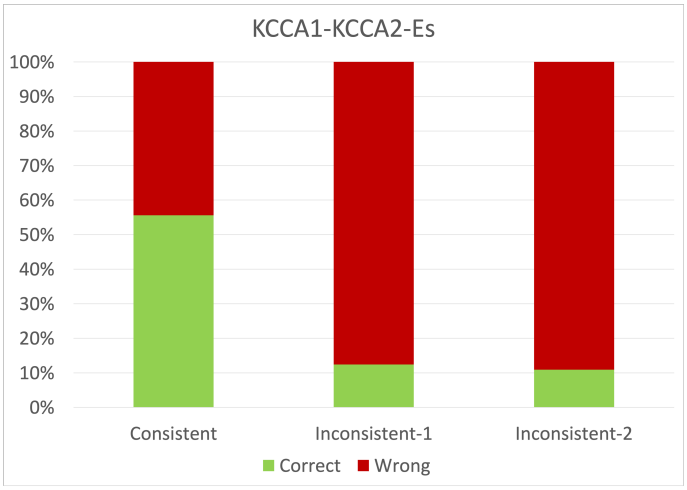



**(b)** CCA's self-consistency.

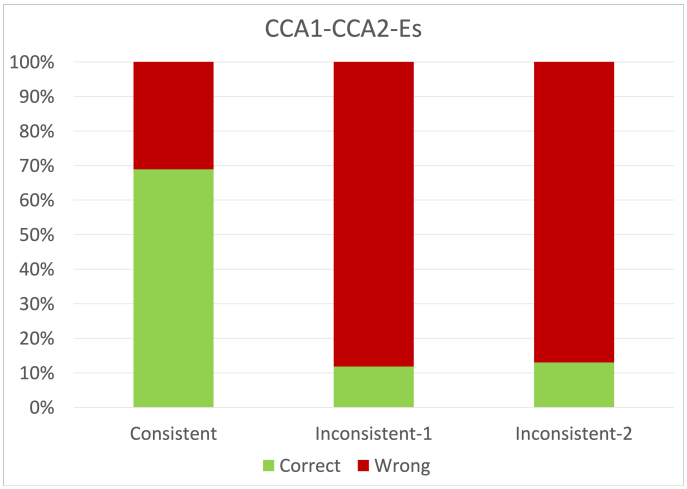



**(c)** Vecmap's self-consistency.

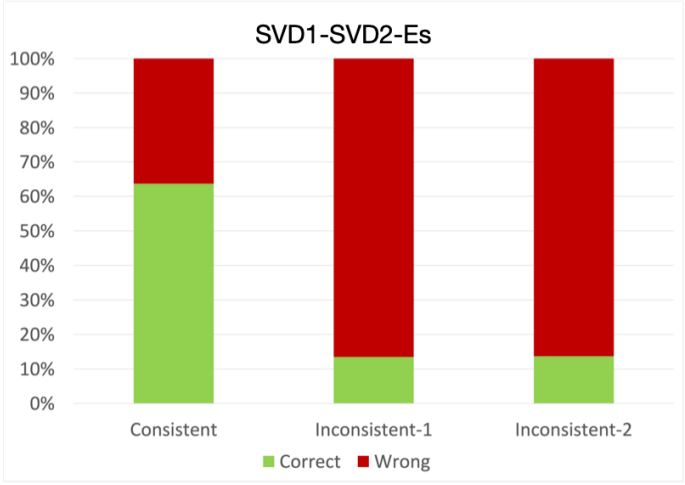
**Figure 42:** Self-consistency on the En-Zh dataset. '1' and '2' means two training subsets.




**(a)** The self-consistency in KCCA-based mapping.




**(b)** The self-consistency in CCA-based mapping.




**(c)** The self-consistency in SVD-based mapping.

**Figure 43:** Self-consistency of different mapping-based methods on the En-Es dataset. '1' and '2' means two training subsets.

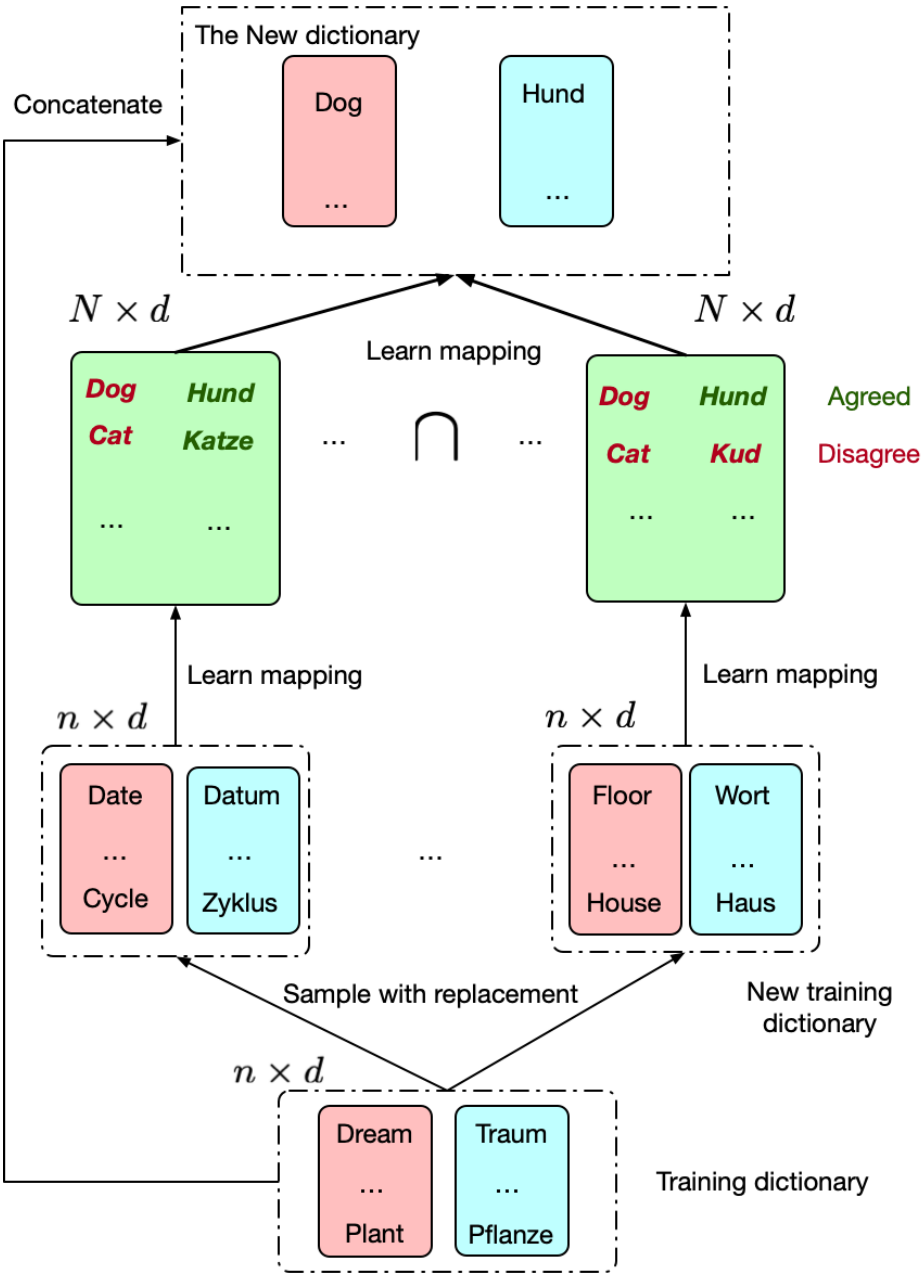



**Figure 44:** An example of my proposed consistency-based model between English and German words.

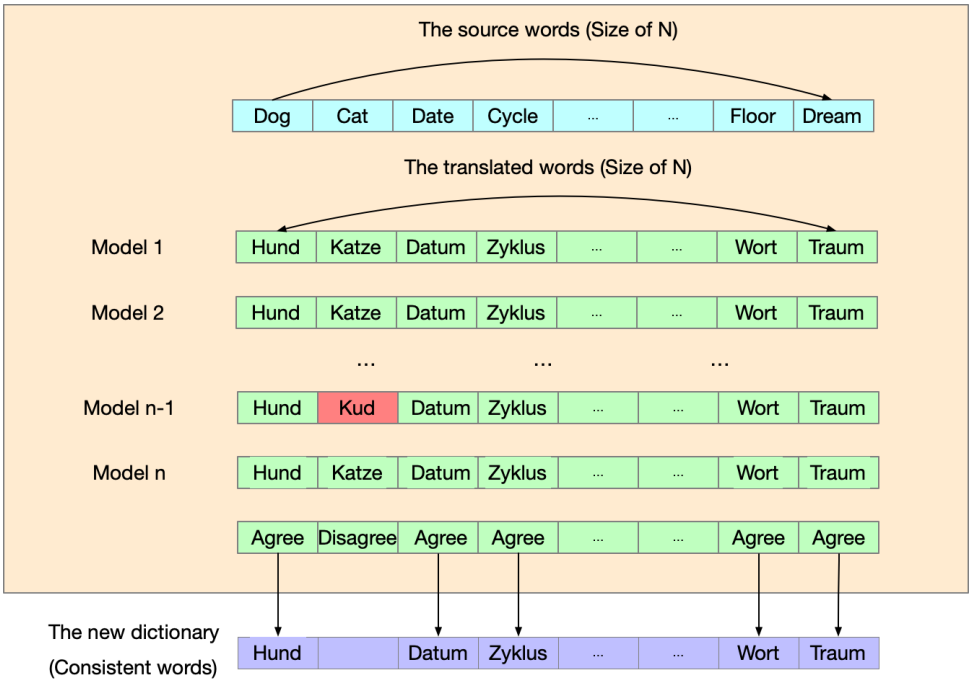



**Figure 45:** An illustration of the model agreement process.

indicates that the agreement model is more robust and achieves better results compared with a single model trained by the original training dictionary.

**Algorithm 4** Agreement model

Dictionary $d$, monolingual word embeddings $X_s$, $X_t$

**repeat**

1. Sample $m$ entries from $d$, obtain word embedding matrices $x^i, y^i (i \in (1, I))$.
2. Find projection matrices $W_x$ and $W_t$.
3. Obtain bilingual word embeddings $\tilde{X}_s^i$, $\tilde{X}_t^i$.
4. Obtain similarity matrix $M^i$ using a similarity metric described in Equation 5.1.
5. Generate a new dictionary $d^i$ with size $R$ by calculating the topK similar nearest neighbours:

$$d^i = \text{topK}(M_i) \tag{5.2}$$

**until** $i == I$

6. Calculate the intersection of all $L$ models and origin dictionary $d$ and cut it to size $r$:

$$d_{union} = d^1 \cap d^2 \cap ... \cap d^L \cap d \tag{5.3}$$

7. Find the final projection.

However, the preliminary experiments show Algorithm 4 is not good enough to find the best local optima. Therefore, I next propose a further investigation, aiming to improve Algorithm 4.

- **RANSAC Initialisation**. The proposed model is affected by outliers in both the training dictionary and the word embedding matrices. The words that do not appear

typically have a strong influence on the model. Therefore, I propose to incorporate Random Sample Consensus (RANSAC) into the dictionary initialisation process. RANSAC is a parameter-ensemble method that can mitigate the influence of the outliers on the model. The training dictionary sampling process induces noise. The original training dictionary is noised with mismatched word pairs. Then some word pairs are sampled from this noised dictionary, and a new dictionary $d_new$ can be learned with this training dictionary. Then this process iterates multiple times and finds the best-learned dictionary. Subsequently, the union of this dictionary and the original training dictionary is used as the final dictionary to learn the mapping. The Algorithm 5 summarises the proposed idea.

- **Mutual Nearest Neighbours**. To better deal with the Hubness (See Section. 3.6.3), I proposed a new dictionary induction method called the *Mutual Nearest Neighbour*. This approach considers the nearest neighbour not from one direction (from source to target), but both directions. Suppose a word embedding pair $(x_s, x_t)$ is the nearest neighbour from the source to the word embedding matrix and from the target to the source direction. In that case, the word embeddings $(x_s, x_t)$ are the mutual nearest neighbours.

- **Frequency-based Dictionary Cutoff**. The low-frequency words in the vocabulary are noisier than the high-frequency words (Artetxe et al., 2018b). In practice, the low-frequency words tend to damage the performance of the learned mapping matrix. Therefore, I proposed to limit the dictionary size to the top k frequent word pairs.

## 5.4 Experiments

### 5.4.1 Dataset

The widely known cross-lingual word embedding datasets developed by Dinu and Baroni (2015), together with its subsequent extension by Artetxe et al. (2017), are used in the experiments. The chosen datasets consist of five language pairs. English-German (En-De), English-Italian (En-It), English-Finnish (En-Fi), and English-Spanish (En-Es). Each European language dataset includes 20k, 300-dimensional monolingual word embeddings trained by CBOW. The English, Italian, and German word embeddings are trained on

**Algorithm 5** The word consistency based model with RANSAC initialisation

Dictionary $d$, monolingual word embeddings $X$, $Y$

1. Induce $m$ noise pairs (mismatched pairs) to the original dictionary.

**repeat**

2. Sample $n$ pairs of words from the nosied dictionary.
3. Find projection matrices.

$$W_s^i, r^i, W_t^i = F^i(X_s, X_t) \tag{5.4}$$

4. Obtain bilingual word embeddings $\tilde{X}^i$, $\tilde{Y}^i$ :
5. Obtain similarity matrix $M^i$ using a similarity metric described in Equation 5.1.
6. Generate a new dictionary $d^i$ with size $R$ by calculating the topK similar nearest neighbours:

$$d^i = \text{topK}(M_i) \tag{5.5}$$

7. Evaluate this dictionary using training set.

**until** $i == I$

8. Find the $d^i$ with highest translation accuracy.
9. Calculate the intersection of $d^i$ and the origin dictionary $d$:

$$d_{union} = d^i \cap d \tag{5.6}$$

10. Find final projection.

Wacky crawling corpora. The Finish monolingual word embeddings are trained on Common Crawl, while WMT News Crawl is used to generate the Spanish word embeddings. The dictionaries are built from OPUS (Tiedemann, 2012). For each language pair, there are 1500 test pairs and 5000 training pairs.

### 5.4.2 Experimental Details

Three experiments are designed to evaluate the proposed approaches. The first experiment uses the vanilla settings outlined in Algorithm 4. The second experiment utilizes the vanilla method plus the dictionary induction approach, while the last experiment uses the RANSAC initialisation outlined in Algorithm 5. Two projection metrics are used in the experiments. The linear projection method is used, which is the orthogonal method proposed by Artetxe et al. (2018a) and the non-linear method KCCA, proposed by Zhao and Gilman (2020). For the parameters described in Algorithm 4, the $L = 20, 100, 200, 300$, $m = 5000$ , $R = 40000$ and $r = 20000$ values are used. For the Ransac initialisation, 1000 pairs are randomly re-ordered and set as noise and the remaining 4000 pairs as the ground truth. The experiments were implemented in Python3 using an open source library

CuPy[1]. All experiments were carried out on two 11GB Nvidia 1080 Ti GPUs on a PC with 64 GB RAM. The weight parameters are set in the ensemble method as a hyper-parameter and a grid search algorithm is applied to find the best hyper-parameter set.

The open-source framework Vecmap[2] is utilised to implement the pipeline of the mapping-based cross-lingual word embeddings. It has two settings. The supervised setting and the unsupervised setting. For the supervised setting, Vecmap provides a generalized five-step mapping which contains normalisation, whitening, re-weighting, de-whitening, and dimensionality reduction (Artetxe et al., 2018a). I follow the default settings provided by Artetxe et al. (2018a). Both source and target word embeddings apply length normalisation, mean centering, whiting and de-whitening; the reweighing parameter $\mathrm{r}$ is set to 0.5. There is an inner similarity between re-weighting and dimensionality reduction based on the finding in Artetxe et al. (2018a). Therefore, following this work, I only adopt the re-weighting process and ignore the dimensionality reduction process.

For the unsupervised setting, an iterative two-step procedure is implemented to avoid the necessity for an extensive dictionary. Additionally, an initialisation of a start-up dictionary is created. The linear mapping is first estimated using this dictionary, and then this dictionary is augmented by applying the nearest neighbour of the projected cross-lingual word embeddings. Those two steps repeat until the convergence of a specified criterion. In the proposed experiment, we follow the default setting provided by Artetxe et al. (2018b). The results from both supervised and unsupervised approaches are provided.

The RANSAC initialisation process set the sampling number $\mathrm{n} = 2000$ and the iteration number $\mathrm{I} = 200$. The experiments on RANSAC are on the En-Es and En-It datasets.

### 5.4.3 Ensemble Model

The naive bagging-based ensemble is adapted for the pipeline of the mapping-based method (Orthognal method). For this, I first define $\mathrm{m} = (2, 3, 4, 5)$ models. In each model where $\mathrm{n} = (2500, 3000, 3500, 4000, 4500)$, the training dictionaries are first sampled and applied to the linear mapping.

There are naturally two ensemble aggregation strategies that can be applied:

1. Average similarity. This is the most used ensemble strategy (Tsoumakas and Vlahavas, 2007). Under this approach, during the dictionary induction process, the

[1] https://cupy.dev/
[2] https://github.com/artetxem/vecmap

similarity matrix of both models is averaged and this averaged similarity matrix is then used to find the nearest neighbours of a specific word.

2. Average rank. After the dictionary induction process, a list of rankings of the similarities of each word can be then be obtained. For instance, given a word embedding $w_s^i$, first, calculate the similarity between $w_s^i$ and word embeddings in the target word embedding matrix $X_t$ and obtain a similarity vector $w_s^i X_t$. Then formulate a rank order list based on this similarity vector. Based on this process, two rank order lists can be obtained using two mapping-based approaches, such as KCCA and the orthogonal method. This strategy averages those two rank order lists and selects the nearest neighbours as the top one of this averaged rank list.

## 5.5 Result Analysis

### 5.5.1 Comparing to the Pipeline of Mapping-based Method

**Table 5.1:** The size of the learned dictionary in the proposed model.

| | Learned Dictionary | Union Dictionary | Cut Dictionary |
|---|---|---|---|
| En-It | 16573 | 18923 | 18774 |
| En-De | 15222 | 17895 | 17529 |
| En-Es | 14324 | 16784 | 16529 |
| En-Fi | 11353 | 14680 | 13687 |
| En-Zh | 11392 | 17653 | 16584 |

Table 5.1 shows the size of the augmented dictionary is larger than the size of the original dictionary (the original dictionary size is 5000). The dictionary augmentation approach can provide additional training data in an unsupervised scenario. This experiment aims to evaluate the performance of the augmented dictionary. Table 5.2 confirms that the proposed augmentation approach has a better performance in all language pairs than other individual mapping-based methods. The proposed model achieves 0.9, 3.1, 1.0, 1.7 percentage point improvements in terms of word translation accuracy on the En-Es, En-De, En-It, and En-Fi datasets respectively. This result indicates that the proposed dictionary augmentation process is more efficient than the original dictionary using the same orthogonal model.

**Table 5.2:** A comparison of results (accuracy) of the proposed model.

| **Method** | **En-Es** | **En-De** | **En-It** | **En-Fi** | **Average accuracy** (%) |
|---|---|---|---|---|---|
| Unsupervised | 37.3 | 48.1 | 48.2 | 32.6 | 41.6 |
| Supervised | 38.2 | 47.2 | 47.3 | 35.0 | 41.9 |
| The proposed | | | | | |
| Ensemble | **39.5** | 48.7 | **48.5** | 36.3 | 43.3 |
| Proposed model | 39.1 | **50.3** | 48.3 | **36.7** | **43.6** |

### 5.5.2 Compare to Naive Bagging

Table 5.2 shows that the proposed approach outperforms the ensemble method by 1.6, 0.4 percentage points in the En-De and En-Fi datasets. Additionally, Table 5.2 shows the results of the two baseline models (supervised and unsupervised). Both ensemble models outperform the single mapping-based model. The result confirms one of my hypotheses that the training dictionary contains noises and redundancy. The findings indicate that both ensemble models can effectively alleviate the impact of the outliers and provide robust and more accurate results.

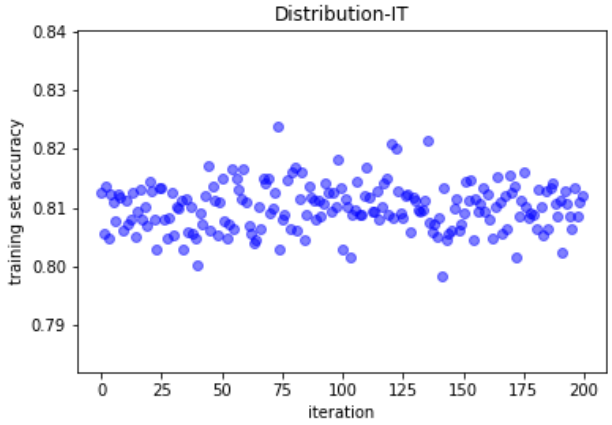
### 5.5.3 Analysis of the RANSAC Initialisation

**Table 5.3:** The word translation accuracy comparison of the RANSAC Initialisation and the vallina mapping-based method.

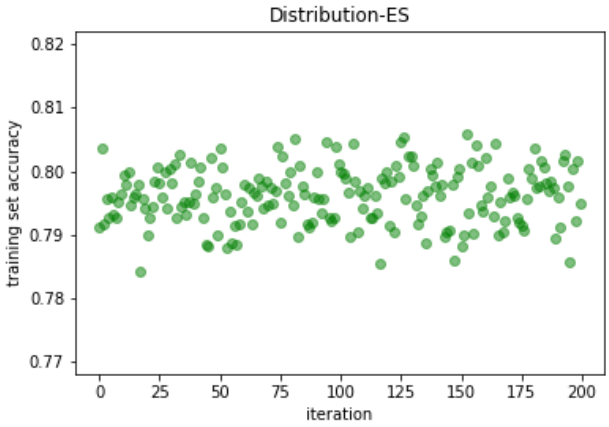
| **Method** | **En-Es** | **En-It** |
|---|---|---|
| Training set result | | |
| RANSAC (best result) | 80.59 | 82.39 |
| Test set result | | |
| RANSAC | 37.73 | 47.20 |
| Orthogonal | 38.20 | 47.30 |
| Consistency | 39.13 | 48.26 |

Figure 46 shows the track of the training process of Algorithm 5. In the experiment, I record the training set word translation accuracy in each loop. The result shows there is no visible relationship between the training iteration and the word translation accuracy.

Table 5.3 illustrates the adoption of the RANSAC initialisation does not substantially impact the word translation accuracy of the test sets - it slightly degrades the performance of the learned mapping. The result has is 0.9, 0.1 percentage points drop than the orthogonal method on En-Es, En-It test set.




**(a)** The training set accuracy distribution on the En-It dataset.




**(b)** The training set accuracy distribution on the En-Es dataset.

**Figure 46:** The accuracy evaluated on training set using the learned dictionary.

### 5.5.4 Analysis of the Mutual Nearest Neighbours Dictionary Induction

The proposed experiment aims to investigate whether mutual nearest neighbours could provide improvements to the final results. For simplicity, the mapping process and the dictionary induction process are only executed once, which is the same with the pipeline of linear mapping (See Section.3.4). I only changed the dictionary induction process to the proposed mutual nearest neighbours process. The result is illustrated in Table 5.4.

**Table 5.4:** Word translation accuracy (%) of different dictionary induction process.

| | **En-It** | **En-De** | **En-Es** | **En-Fi** | **En-Zh** | Average accuracy |
|---|---|---|---|---|---|---|
| Nearest neighbours | 47.46 | 47.46 | **39.00** | 36.16. | 49.14 | 43.90 |
| Mutual nearest neighbours | **48.06** | **49.00** | 38.53 | **36.65** | 49.00 | **44.25** |

The proposed dictionary induction process provides competitive results on the En-It, En-Es and En-Fi datasets. Table 5.4 shows the mutual nearest neighbours dictionary induction gains an averagely 0.35 percentage point increase in terms of word translation

accuracy. The result confirms that the mutual nearest neighbour can effectively perform a dictionary induction process that aims to find a better dictionary.

### 5.5.5 Analysis on the Frequency-based Dictionary Cutoff Strategy

Table 5.5 shows that the frequency-based dictionary-pruning process provides a better result than simply using the learned dictionary. This is possibly due to the outliers being mostly distributed amongst low-frequency words. The dictionary cutoff process provides an average 0.1 percentage point improvement across all datasets in terms of word translation accuracy.

**Table 5.5:** Word translation accuracy (%) of baseline and the model with the dictionary-pruning process.

| | **En-It** | **En-De** | **En-Es** | **En-Fi** | **En-Zh** | Average accuracy |
|---|---|---|---|---|---|---|
| Baseline | **47.73** | 50.80 | 39.33 | **37.00** | 49.14 | 44.80 |
| Dictionary Cutoff | **47.73** | **50.87** | **39.40** | 36.86 | **49.61** | **44.89** |

However, some low-frequency words can contribute to the final projection. The dictionary cutoff process harms the proposed model's performance in the En-Fi dataset, causing a 0.14 percentage point drop. This accuracy decrease is caused by the data insufficiency issue found in the En-Fi dataset. The En-Fi dataset is much smaller than rich-resource datasets like En-It and En-Es. Therefore the word embeddings appear to be more sensitive to the inadequate information. The trade-off between the size of the dictionary and the quality of the cross-lingual word embeddings still needs to be further studied.

### 5.5.6 Model Number's Impact

This experiment develops the relationship between the dictionary quality and the weak model size in the augmentation approach. The word translation accuracy of the training set is used to evaluate this relationship. Figure 47 shows that when the number of the weak models rises, the translation accuracy of the training pairs also improves. The word translation accuracy increases to 80% when 20 models are combined. The result illustrates that an ensemble of more models provides a higher translation accuracy in the test set. The result confirms the above mentioned hypothesis that the multiple weak models offer diversity, and an ensemble of them can generate a robust and better local optima.

Figure 48 indicates that the correctly translated word pairs decrease when more models are incorporated. The result confirms the hypothesis in Section 5.2 that the consistent word embeddings are strongly correlated and share similar geometric arrangements. Ad-

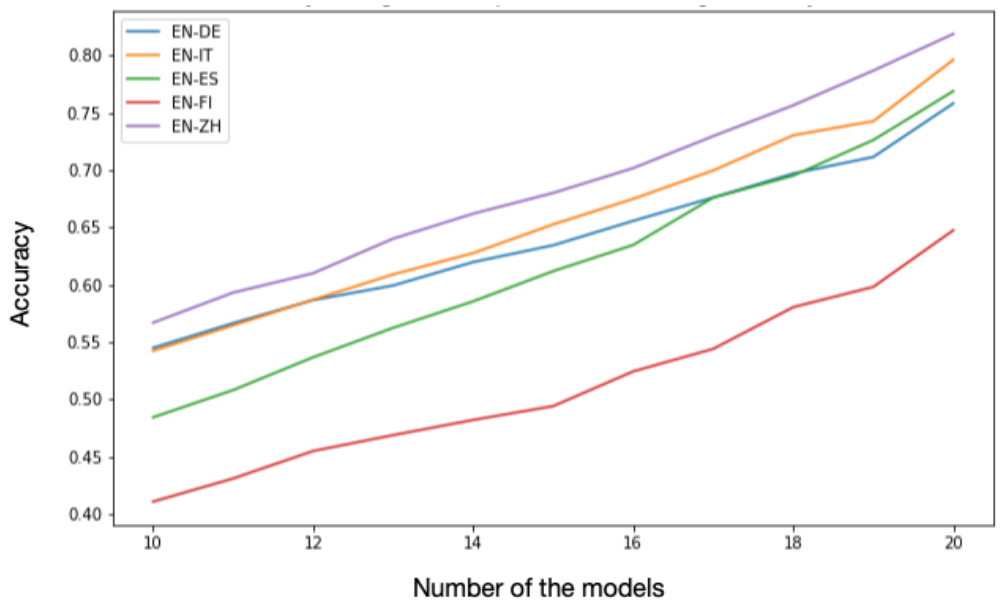



**Figure 47:** The word translation accuracy of training word pairs with a different number of models.

ditionally, in the experiment, only the word pairs agreed by all sub-models can be seen as consistent word pairs. This strong constraint, also called "unanimity", often applies in ensemble medical models like heart attack prediction (Raza, 2019), which can generate the strongest and the most reliable model. Therefore, those corresponding word pairs can be considered as a reliable training dictionary.

### 5.5.7 Analysis of the Ensemble Strategy

Table 5.6 manifests the result using different ensemble strategies. The average strategy works better on the En-Fi dataset, and the average rank strategy works better on other datasets. In total, the rank strategy works better for ensemble models. This is probably because the ranking strategy only focuses on the significance of each word and disregards the effect of real features, making the model more robust.

**Table 5.6:** Word translation accuracy (%) of different ensemble strategies.

| | **En-It** | **En-De** | **En-Es** | **En-Fi** |
|---|---|---|---|---|
| Average Similarity | 47.80 | 47.78 | 39.13 | **36.86** |
| Rank | **48.26** | **49.60** | 39.13 | 36.44 |

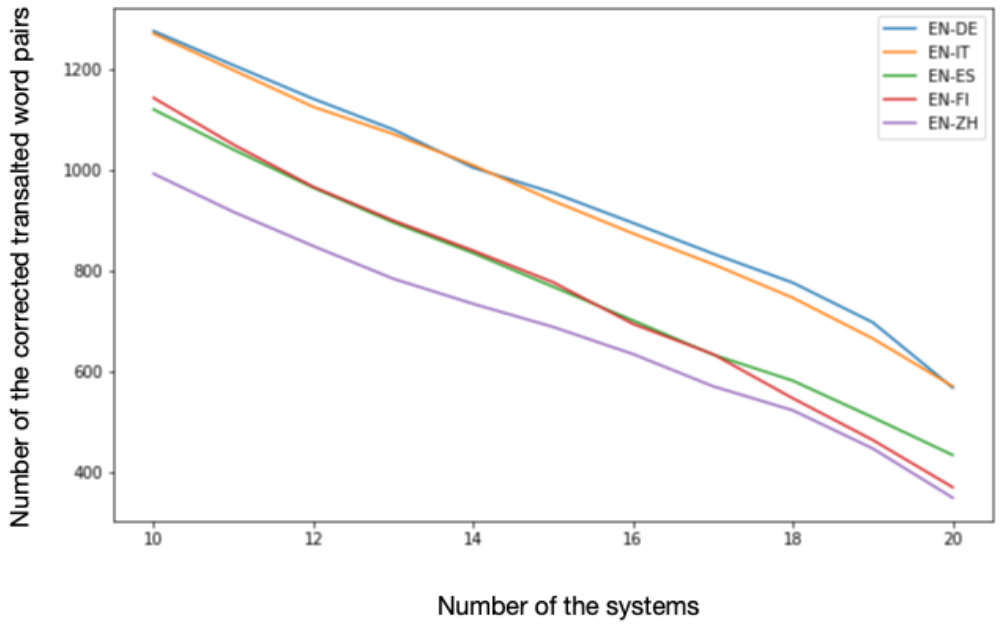



**Figure 48:** Number of the correctly translated word pairs within consistent word pairs when different number the models are combined.

## 5.6 Conclusion & Future work

An essential challenge of the cross-lingual word embedding approach is the limitation of the training dictionary. The noise of the training dictionary leads to the high variance issue.

In this chapter, I introduce my investigation on dealing with the issue mentioned above. First, I adapt the ensemble model to the cross-lingual word embedding approaches. The proposed method ensembles the orthogonal and KCCA models to leverage the diversity in linearity and non-linearity. Additionally, to address the inadequate data issue in the training dataset, I introduce a word consistency-based dictionary augmentation method that improves the mapping-based cross-lingual word embeddings by enlarging the training dictionary in a semi-supervised scenario. I further develop the effect of the detailed strategies of the proposed model, including the RANSAC initialisation, the frequency-based dictionary pruning and the rank-based ensemble strategy.

I designed extensive experiments to evaluate the proposed approach. The performance of the bagging-based ensemble model is better than the individual models, and the proposed consistency-based ensemble model also provides better results. The result confirms two hypotheses mentioned in this chapter: 1. The training dataset contains noise, and the noise has a negative affecting on the learned mapping. 2. The diversity of different weak

models contributes to the final ensembled model.

In the investigation of the RANSAC initialisation, the result shows the strategy does not benefit the proposed model. I also experimented with the frequency-based dictionary cutoff strategy, which helps mitigate the outliers' adverse effects. Additionally, the experimental result confirms that the rank-based process can effectively ensemble the two models.

CHAPTER 6

# Mutual Learning for Speech Translation

A popular research area in end-to-end speech translation is distilling knowledge from a machine translation model to a speech translation model. This transfer paradigm views a trained machine translation model as the teacher and a non-trained speech translation model as the student. A limitation in this approach, however, is that it only allows for the one-way transfer, which raises two issues and these two tasks can not be accommodated into a teacher-student paradigm.

1. **The performance of the teacher framework limits the one-way transfer paradigm.** The one-way transfer paradigm is based on the assumption that the performance of the teacher model outperforms the student model so that the teacher model can guide the student model in the training process. Therefore, the performance of the teacher model highly impacts the performance of the student model. However, the teacher model limits the student model by fixing the training target, which leads to a constraint on the search space of the student model.

2. **The different nature of voice and text input makes speech and machine translation naturally fit with different models.** In an ideal teacher-student scenario, the teacher and student models are trained to solve identical tasks. However, under the circumstance in this chapter, the teacher model (MT) and the student model (ST) have a considerably different structure and are not comparable.

In order to address the issues mentioned above, a trainable mutual learning-based end-to-end speech translation paradigm is designed instead of conventional one-way training. The proposed training paradigm effectively improves the translation performance of the

end-to-end speech translation model. The proposed model contains two separate training components: an end-to-end speech translation model and a machine translation model. Additionally, a proposed mutual learning scenario trains those two models collaboratively. This simple training paradigm improves the performance of both models. Experimental results show that the proposed model can efficiently explore the auxiliary information from peer models and improve both of them. In the same setting, the proposed model exceeds the baseline model by on the En-Fr, En-Es, and the En-De MuST-C datasets by 2.4, 1.8, 0.1 BLEU score, and the best result achieves state-of-the-art in this field.

## 6.1 Introduction

Speech translation (ST) aims to translate speech signals into a foreign language. It is a multi-modality task, closely related to automatic speech recognition (ASR) and machine translation (MT). ST has a wide range of applications, such as video subtitling (Saboo and Baumann, 2019), real-time lecture translation (Müller et al., 2016), and protection of endangered languages (Bansal et al., 2017).

Despite recent successes in end-to-end (E2E) models, currently, those systems still face the issue of data insufficiency (Sperber and Paulik, 2020). A widely used recent advance in E2E ST is knowledge distillation (KD), which provides an effective paradigm for transferring knowledge from rich-resource to low-resource tasks (Liu et al., 2019; Gaido et al., 2020b). These models consider the MT model as teacher to guide the ST model, which is regarded as a student. I hypothesise that a strict teacher-student scenario may be sub-optimal for the following reasons:

- The MT model freezes (only used in an inference scenario, meaning that the model is not jointly trained) in this one-way knowledge transfer scenario, the success of knowledge transfer and hence the performance of the ST task is constrained by the performance of the pre-trained MT model (See details in Section 6.2.4).
- There exists a modality gap between speech and text inputs of the two models, with speech input also containing inherent speaker variability.

Motivated to address the issues mentioned above, I set out to improve ST and MT tasks by training them jointly. Instead of freezing the teacher model, a mutual-learning paradigm is proposed, which regards ST and MT models as peers that learn collaboratively, aiming to share the knowledge between the two models iteratively. Mutual learn-

ing has been proposed to leverage information from multiple models and allows effective dual knowledge transfer in image processing tasks (Zhang et al., 2018; Zhao et al., 2021a). This idea is leveraged and adapted to sequence tasks. This chapter describes the main contributions listed below:

- I propose a jointly-trainable mutual-learning paradigm, which improves the distillation method by training them together. MT and ST's search spaces are enlarged, providing the potential for more robust local optima.
- I further improve the proposed mutual-learning method by integrating the cyclical annealing schedule, which alleviates the KL vanishing problem from which many time-series tasks suffer. (Fu et al., 2019; Bowman et al., 2016; Higgins et al., 2016)
- Extensive experiments on MuST-C En-Fr, En-Es datasets are implemented. The experimental results illustrate the advantage of the proposed model by empirically comparing it with a cascaded model, a knowledge distillation (KD) model and a multi-task learning (MTL) model. The experimental results show that the proposed model can effectively leverage the transcript (source speech text) and the auxiliary MT task and obtain competitive results in all experiments. In addition, as a side benefit, the performance of the MT model also improves.

## 6.2 Background

### 6.2.1 Definition of Speech Translation

Given the speech feature $\mathsf{X}$ from one language, and corresponding ground truth translation text $\mathsf{T}$ from another language, the speech translation model aims to maximise the posterior probability of $\mathsf{X}$ given $\mathsf{T}$:

$$\hat{\mathsf{T}} = \underset{\mathsf{T}}{\operatorname{argmax}}\, \mathsf{P}(\mathsf{T}|\mathsf{X}) \tag{6.1}$$

### 6.2.2 Cascaded Model

Earlier cascaded models integrate two loosely coupled models: an ASR model and a MT model. In cascaded models, the speech features and their corresponding source texts are first used to train an ASR model. The ASR model can transfer source speech features to source speech texts (also called transcripts). Additionally, an MT model is followed

to transfer the source speech texts into target speech texts. Figure 49 illustrates cascaded model.


**Figure 49:** Cascaded model.

Based on Bayes' theorem, the Equation 6.1 can be reformulated by Equation 6.2:

$$\hat{T} = \underset{T}{\operatorname{argmax}}\, P(T|S,X)P(S|X) \tag{6.2}$$

In Equation 6.2, S represents the corresponding source transcript. The second component can be seen as an ASR model $P_{ASR}(S|X)$. Given a speech feature, the ASR model provides the hypothesis of its transcript texts. In general, the first component can be roughly seen as an MT model (Sperber and Paulik, 2020).

$$P(T|S) \approx P_{MT}(T|S,X) \tag{6.3}$$

Therefore Equation 6.1 can be decomposed as:

$$\hat{T} = \underset{T}{\operatorname{argmax}}\, P_{MT}(T|S)P_{ASR}(S|X) \tag{6.4}$$

where $\hat{T}$ is the hypothesis for the final translation of the cascaded model.

The cascaded model has two challenges:

1. The ASR Model inference result is not reliable enough to be passed to the MT model.

2. Source language mismatches between ASR and MT model. ASR's training data contains more informal language texts and interjection words, and those data often come from daily spoken languages; the MT training data contains more formal language texts like news and reviews as well as Wikipedia articles.

Due to the above, recent research efforts have started to focus on direct models which try to build a direct model using Equation 6.1.

### 6.2.3 The Vanilla Model of End-End Speech Translation

E2E ST learns a single model which directly maps speech features to a target language text sequence (Weiss et al., 2017; Duong et al., 2016). Given a sample pair $(x, y)$ from

the training set D corresponding to speech signal and translated target sentence, the ST model is trained by minimising:

$$L = -\sum_{(x,y)\in D} \log P(y|x;\theta) \tag{6.5}$$

where L is the negative log likelihood (NLL) loss. E2E models consist of an encoder that encodes speech input as an intermediate representation, and a decoder that decodes this intermediate representation to a probability distribution over the target text feature space. Previous work on E2E ST have been developed using recurrent neural network-based models, but since has moved on to Transformer-based models (Gangi et al., 2019b; Zhang et al., 2019; Vila et al., 2018; Weiss et al., 2017; Berard et al., 2016). Figure 50 shows the general framework for end-to-end speech translation models.

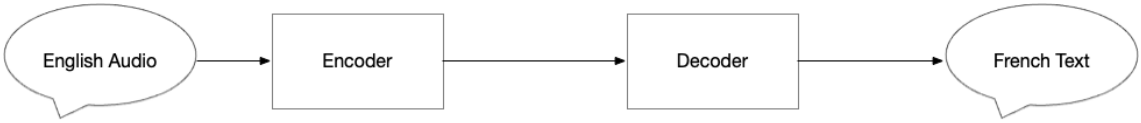



**Figure 50:** The end-to-end speech translation model.

### 6.2.4 The Knowledge Distillation for End-to-End Speech Translation

A typical teacher-student framework is knowledge distillation (Hinton et al., 2015), which transfers knowledge from a large model to a smaller model. The knowledge distillation approach is widely used in model compressing, knowledge transfer and dealing with data scarcity issues (Zhao et al., 2021a; Liu et al., 2019; Zhang et al., 2018).

MT model is generally used as a teacher model to transfer knowledge to the speech translation model to deal with data scarcity. Knowledge distillation loss consists of the re-construction loss and the distillation loss. Denote $L_{st}$ as the log-likelihood loss for the ST model:

$$L_{st} = -\sum \log P(y|x;\theta) \tag{6.6}$$

where parallel data $(x, s, y)$ comes from speech feature X, transcript S and target text Y, and the $L_{st}$ is the reconstruction loss.

In the distillation model, the distillation loss $L_{kd}$ is normally the cross-entropy between the output distributions of the ST and MT model. Given a token $y_i$, the distribution can

be expressed as $Q(y_i|y_{<i}, x)$. Then the distillation loss can be defined:

$$L_{kd} = \sum_{i=1}^{N} \sum_{k=1}^{|V|} Q(y_i = k|y_{<i}, x; \theta_q) \log P(y_i = k|y_{<i}, x; \theta) \tag{6.7}$$

where $\theta$ is the parameter set of the student model and $\theta$ is the parameter set of the teacher model. Finally, the loss of the knowledge distillation model L is defined as:

$$L = (1 - \lambda)L_{st} + \lambda L_{kd} \tag{6.8}$$

The MT model is pre-trained and frozen in the knowledge distillation scenario for the end-to-end speech translation. During the training process, the distribution Q is obtained by the inference mode of the machine translation model. Then the knowledge can be transferred from the MT to ST model through this scenario. As discussed before, the KD scenario is challanging to adopt between the ST and MT models because of the modality gap and the performance constraint of the MT model.

## 6.3 Proposed Mutual-Learning-based Speech Translation

### 6.3.1 Model Description

**Model definition:** Given parallel data $(x_i, s_i, y_i)$ from speech feature X, transcript S and target text Y, and an ST model $M_{st}$ and an MT model $M_{mt}$, the output probabilities are given by:

$$p_{st} = M_{st}(x_i) \tag{6.9}$$

$$p_{mt} = M_{mt}(s_i) \tag{6.10}$$

The training loss has two components: a traditional supervised reconstruction loss and a mimicry loss that aligns the output posterior distributions between the models. The Kullback–Leibler (KL) divergence (Kullback and Leibler, 1951) is adapted as the mimicry loss, aiming to reduce the distance of outputs of ST and MT systems, effectively encouraging them to mimic each other. Since KL divergence is asymmetric. It is calculated in both directions as follows:

$$KL_1 = KL(p_{mt}||p_{st}) = \sum_{j}^{N} p_{mt}^{j} \ln \frac{p_{mt}^{j}}{p_{st}^{j}} \tag{6.11}$$

$$KL_2 = KL(p_{st}||p_{mt}) = \sum_{j}^{N} p_{st}^{j} \ln \frac{p_{st}^{j}}{p_{mt}^{j}} \tag{6.12}$$

where N represents the length of the output sentence. The NLL loss is used as the reconstruction loss, denoted by $LC_{st}$ for ST and $LC_{mt}$ for MT:

$$LC_{st} = -\sum_{i}^{N} y_i \ln (p_{st}^{i}|y_i) \tag{6.13}$$

$$LC_{mt} = -\sum_{i}^{N} y_i \ln (p_{mt}^{i}|y_i) \tag{6.14}$$

Lastly, a weighting term is assigned to the mimicry loss and combined with the reconstruction losses to produce the mutual learning loss, as described by Equation 6.15:

$$L_{ml} = \beta(KL_1 + KL_2) + LC_{st} + LC_{mt} \tag{6.15}$$

Figure 51 illustrates the proposed mutual learning scenario.

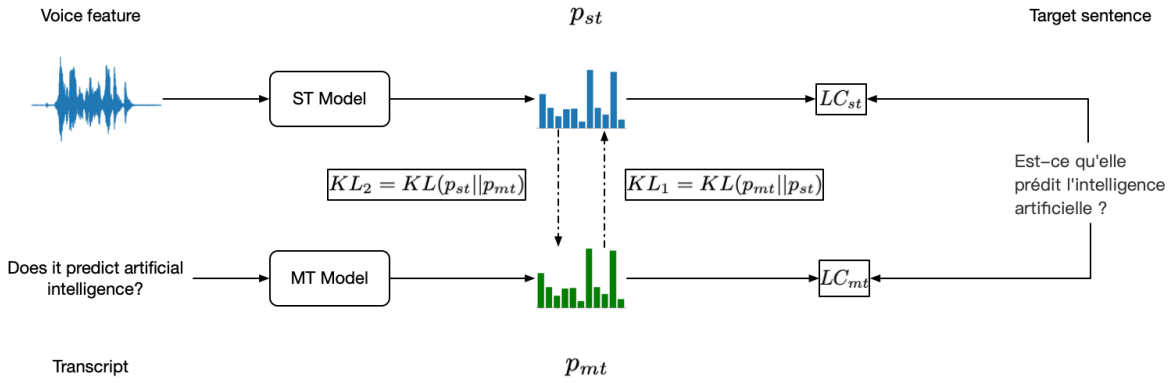



**Figure 51:** The proposed deep mutual learning scenario. The training objective contains four separate components, the reconstruction losses of ST and MT ($LC_{st}$ and $LC_{mt}$) and KL divergence between outputs of ST and MT ($KL_1$ and $KL_2$).

### 6.3.2 Training Strategy

As shown in Algorithm 6, ST and MT models are trained iteratively until convergence. In each iteration, there are two steps: 1. the MT model is frozen and the parameters of ST model are updated; 2. ST model is frozen and the parameters of MT model are updated. Figure 51 illustrates the training process.

**KL vanishing issue**: In the proposed mutual learning strategy, leveraging KL divergence as a loss suffers from the vanishing issue, which can be alleviated by adopting a cyclical annealing schedule for $\beta_t$. The cyclical annealing schedule was proposed to mit-

**Algorithm 6** Training Strategy

Input: training set, ST network parameters $\theta_{st}$ (with ASR pre-trained encoder), pre-trained MT network parameters $\theta_{mt}$

**repeat**

$t = t + 1$

1. Compute $p_{st}$ and $p_{mt}$ for one minibatch
2. Freeze $\theta_{mt}$, compute the gradient and update $\theta_{st}$

$$\theta_{st} \longrightarrow \theta_{st} + lr * \frac{\partial L_{ml}}{\partial \theta_{st}} \tag{6.16}$$

3. Upate $p_{st}$ and $p_{mt}$
4. Freeze $\theta_{st}$, compute the gradient and update $\theta_{mt}$

$$\theta_{mt} \longrightarrow \theta_{mt} + lr * \frac{\partial L_{ml}}{\partial \theta_{mt}} \tag{6.17}$$

**until** convergence

igate similar KL vanishing issue in variational auto-encoders (Fu et al., 2019). In this case, $\beta_t$ in Equation 6.15 changes periodically during training iterations, as described by Equation 6.18:

$$\beta_t = \begin{cases} \frac{r}{RC}, & r <= RC \\ 1, & r > RC \end{cases} \tag{6.18}$$

where $t$ represents the current iteration and $r$ is defined as:

$$r = mod(t-1, C) \tag{6.19}$$

The training process is effectively split into many cycles with each cycle lasting $C$ iterations. In each cycle $\beta_t$ progressively increases from 0 to 1 during $RC$ iterations and then stays at 1 for the remaining $(1-R)C$ iterations. Also, $R = 0.5$ and $C = 5000$. With this trick the model can be trained and the KL vanishing issue can be mitigated.

## 6.4 Experiments

### 6.4.1 Dataset

I evaluate the proposed framework on the popular MuST-C multilingual speech translation dataset[1] (Gangi et al., 2019a), using the two most used language pairs: English-to-French (En-Fr) and English-to-Spanish (En-Es). En-Fr dataset contains 500 hours of training voice clips and 280K pairs of parallel sentences. En-Es dataset contains 504 hours of

[1] https://ict.fbk.eu/must-c/

training voice clips and 270K pairs of parallel sentences. Table 6.1 shows the detail statistic of the two datasets.

**Table 6.1:** The MuST-C dataset statistics used in the experiments.

| Language pairs | En-Fr | En-Es |
|---|---|---|
| Hours of Speech | 492 | 504 |
| Sentence pairs | 280K | 270K |
| Source tokens | 5.2M | 5.3M |
| Target tokens | 5.4M | 5.1M |

**Pre-processing** The same data pre-processing steps are implemented as described in the fairseq speech-to-text framework (Wang et al., 2020). The pre-processing step extracts 80-channel log Mel-filterbank features and removes the training samples that are larger than 3000 frames. For both transcripts and target texts, the newly proposed subword regularisation method proposed by (Kudo, 2018) is employed to build a vocabulary with the size of 8000. A jointly-trained shared vocabulary of size 8000 is also used as an additional experiment.

### 6.4.2 Training Details

A stack of 2 1D convolutional layers (kernel size 5, stride 2) is used for the ST task, followed by 12 Transformer layers of size 2048 as the encoder. Six stacked Transformer layers with size 512 are used as the decoder. For the MT task, 12 stacked Transformer layers with size 2048 are used as the encoder, and six stacked transformer layers with size 2048 are used as the decoder. Evaluation is based on the standard implementation of BLEU score, SACREBLEU (Post, 2018), with a beam size of 5. The maximum number of tokens in each batch is 40000. Figure 52 illustrates the proposed structure.

The experiments were implemented using on open-source library PyTorch[2] and open-source package fairseq[3]. All experiments were carried out on Nvidia A100 GPU (40GB VRAM) on a PC with 128G RAM.

## 6.5 Results and Analysis

### 6.5.1 Cascaded Model Comparison

First, the cascaded model pre-trains a transformer-based E2E ASR model using speech input and English transcripts in the cascaded setting. Then it pre-trains an MT model using

[2]https://pytorch.org/
[3]https://github.com/pytorch/fairseq

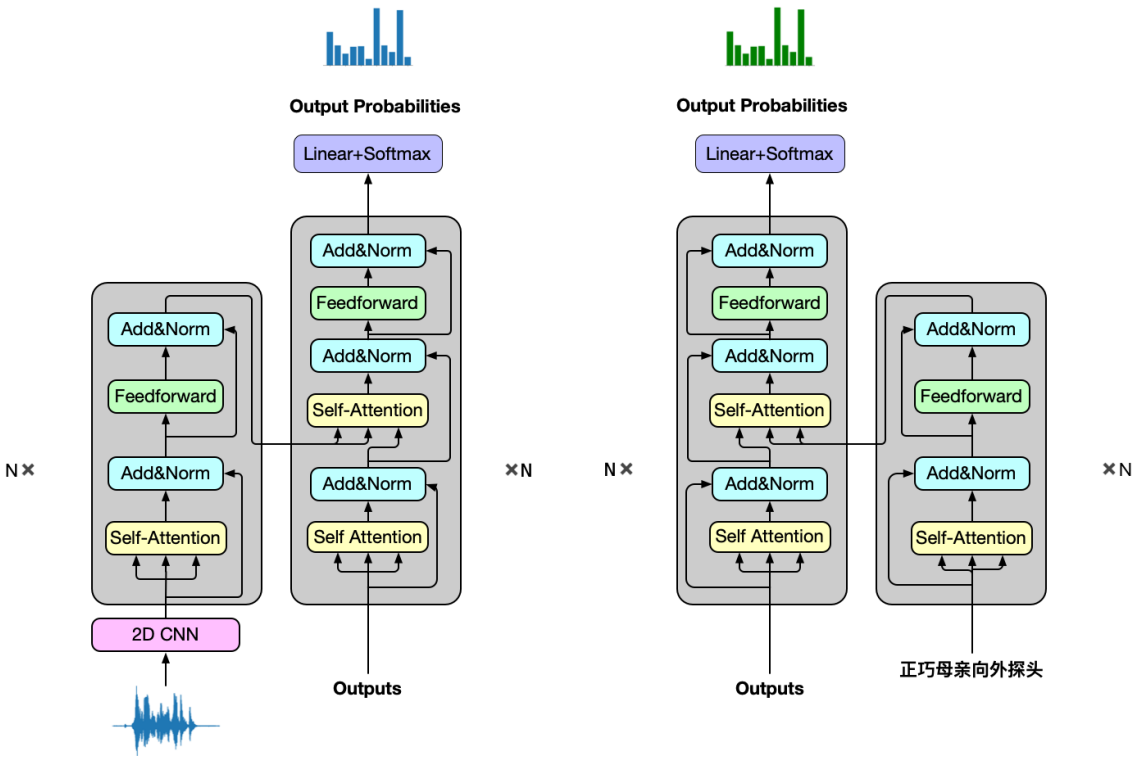



**Figure 52:** Mutual-Learning model detail. The structure of my mutual learning paradigm is two stacked transformer.

English transcripts and target sentences. In inference mode, the ASR model generates an intermediate text representation. The representation is passed to the MT model, and the model calculates and obtains the output probabilities.

As shown in table 6.2, the mutual-learning-based ST model provides competitive results compared to the cascaded model. The proposed model achieves 0.6 and 0.5 BLEU score improvement in En-Fr and En-Es datasets, respectively. The results illustrate that the mutual-learning paradigm provides an effective method for leveraging the additional information available via transcript.

The Vanilla E2E ST model uses different vocabularies for source and target languages. A jointly-trained byte pair encoding (BPE) is also utilised to build the vocabulary to better align those two feature spaces. Leveraging this vocabulary achieves a surprising improvement on the state-of-the-art result, shown in the last row of Table 6.2.

### 6.5.2 Knowledge Distillation Model Comparison

Knowledge distillation (KD) is a conceptually similar approach to the proposed framework. KD provides a one way transfer from a trained teacher model to a student model. The following is a comparison of work in this study with the KD-based method. An MT model is pre-trained using transcripts and target sentences. Then the model is frozen and

**Table 6.2:** A comparison of results for different ST models: Cascaded, vanilla end-to-end, end-to-end with multi-task learning, end-to-end with knowledge distillation, and end-to-end with mutual learning. "⋆" denotes training with a joint vocabulary.

| Method | En-Fr | En-Es |
|---|---|---|
| Cascaded | 34.9 | 28.0 |
| E2E | 32.8 | 27.2 |
| E2E + MTL | 33.5 | 27.5 |
| E2E + KD | 34.5 | 27.9 |
| E2E + ML | 35.5 | 28.5 |
| E2E + ML⋆ | **36.3** | **28.7** |

used to guide an ST model by minimising Equation 6.20:

$$\text{Loss} = \beta * (\text{KL}_1 + \text{KL}_2) + \text{LC} \tag{6.20}$$

where $\text{KL}_1$ and $\text{KL}_2$ are described by Equations 6.11 - 6.12 and LC is the reconstruction loss (Equation 6.13). The difference between KD and the proposed approach is that MT model is frozen and used in an inference mode and only the ST model parameters are updated during the training process. Table 6.2 illustrates the proposed mutual training outperforms one way training strategy by 1.0, 0.6 BLEU score in En-Fr, En-Es dataset, respectively.

### 6.5.3 Multi-Task Learning Model Comparison

Multi-Task Learning (MTL) is also a collaborative learning strategy. In contrast to the proposed mutual-learning scenario, the MTL trains the ST model and MT model separately with the average of the NLL loss from MT and ST models:

$$\text{Loss} = \frac{1}{2} * (\text{LC}_{\text{st}} + \text{LC}_{\text{mt}}) \tag{6.21}$$

ST task results of using the MTL scenario are show in Table 6.2. These results show that the mutual-learning scenario is a more effective way of joint learning: gaining 0.7, 0.3 BLEU score increase comparing to MTL in ST task.

MT task results of using the MTL scenario are shown in Table 6.3. It can be observed that the mutual-learning scenario is more effective for this task as well.

### 6.5.4 Performance of the MT System

In the proposed mutual-learning scenario, the models are trained collaboratively. The ST model and MT model share knowledge through the training process. Therefore, it is interesting to evaluate not only the performance of the ST model, but also the performance

**Table 6.3:** mutual-learning system comparing to independently trained MT system on MuST-C dataset.

| Method | En-Fr | En-Es |
|---|---|---|
| MT | 45.1 | 35.4 |
| MT+MTL | 45.6 | 35.3 |
| MT+ML | **45.8** | **35.7** |

of the MT model. In the proposed experiment, the performance of MT model in the proposed mutual-learning scenario is evaluated and compared with an independently trained MT model which is also trained on MuST-C script-target text pairs. Both MT models have an identical configuration and the same hyper-parameters and training strategy are used, which are described in Section 6.4.2.

From the results in Table 6.3 it can be concluded that mutual-learning also improves the MT model's performance. The proposed model gains 0.7, 0.3 BLEU score in En-Fr and En-Es dataset comparing to the independently trained MT system. The proposed model also exceeds a typical MTL learning model by 0.2, 0.4 BLEU score in the MT task. This result shows that the mutual-learning leads to a more robust minima than that of the MTL paradigm.

### 6.5.5 Analysis of the Cycling Annealing Schedule

Figure 53 illustrates the KL loss, ST loss (reconstruction loss) and the total loss change during the training process on the En-Fr dataset. KL loss gets smaller gradually during the training process. The trend is similar to that of ST loss. This trend indicates that the cycling annealing schedule successfully mitigates the KL vanishing issue, making it possible to use KL distance as the additional mimicry loss to train the system.

## 6.6 Conclusion & Future Works

This chapter discusses the end-to-end speech translation. I review the current approaches of cascaded speech translation and end-to-end speech translation. I describe the key challenge in the speech translation area, which is data insufficiency. I presented my proposed mutual-learning paradigm for end-to-end speech translation to address the low-resource issue. Another contribution described in this chapter is that I addressed the KL vanishing problem by introducing cycling annealing schedule to the training process.

Experimental results demonstrate that the proposed approach outperforms the typical one-way transfer paradigm KD, as well as typical dual knowledge transfer paradigm

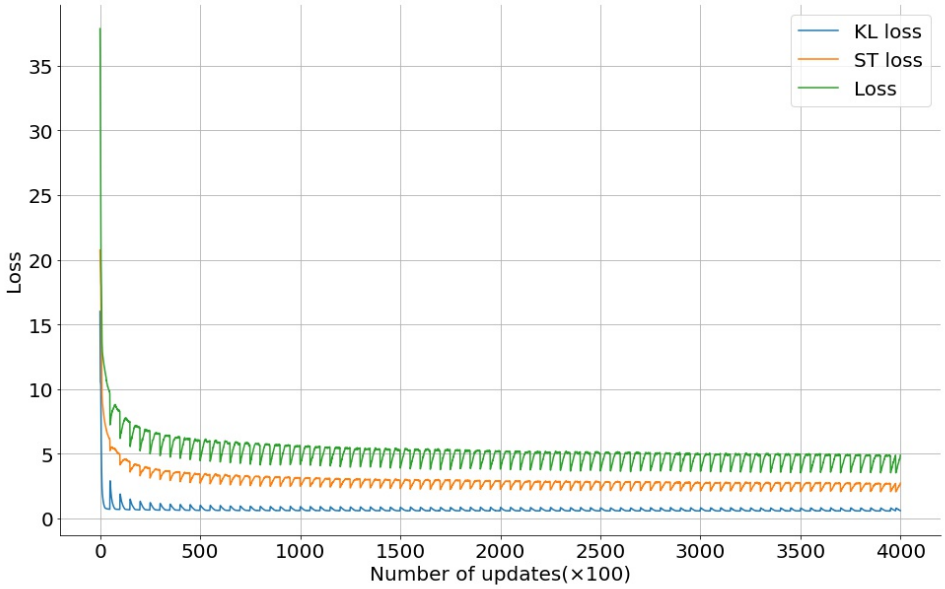



**Figure 53:** The loss for cycling annealing schedule on En-Fr dataset.

MTL. Also, competitive results are obtained compared to cascaded model which has in the past been outperforming E2E ST models. Additional experiments also show that the KL vanishing issue is addressed.

I also further investigated the machine translation model in the proposed scenario. I hypothesised that mutual-learning could benefit both models in the proposed dual knowledge distillation process. The experimental result indicates that MT model can also benefit from the knowledge of ST model.

My future research interest in this area will pursue two topics:

1. The multi-lingual speech translation. Since the inner structure of ST and MT model are similar, it may be possible to design a unified structure that can handle all kinds of languages. This idea has been suggested in MT models but ST models are yet to be covered.

2. Speech translation in extreme low-resource situations. The data insufficiency issue for low-resource languages are hard to deal with. The advances in Meta-learning and Zero-shot learning provide some possible avenues for further exploration.

CHAPTER 7

# Conclusion

In this dissertation, I proposed a series of novel techniques for addressing challenges in the area of cross-lingual word embeddings and speech translation modelling.

In Chapter 3, I demonstrated how my proposed method obtains improvements in mapping-based cross-lingual word embedding methods. In that chapter, I outlined my research into whether non-linearity could better describe the relationships between word embeddings of languages in the different language families which challenges the current hypotheses.

There are several key contributions that emerged from this chapter:

- I proposed two non-linear mapping-based methods, KCCA and DCCA, as the mapping-based methods to learn cross-lingual word embeddings. The experimental work revealed that the proposed approach works better on most datasets than the benchmarked approaches.
- I provided a new Chinese-English dataset which contains the pre-trained word embeddings and a dictionary and corresponding parameter-sets. This dataset is the first Chinese dataset in this area which contains language pairs that are not in the same language family.

In Chapter 4, I proposed my unsupervised cross-lingual word embedding approach. I achieved two objectives:

- In this chapter, I developed the CCA-based mapping for unsupervised cross-lingual word embeddings and proved that CCA-based mapping can also provide competitive results.
- I demonstrated a non-linearity-based unsupervised cross-lingual word embedding approach through a last step of non-linear mapping.

By incorporating KCCA into the mapping process, my proposed method achieves an improvement in results on all language pairs. The improvement on the English-Chinese dataset is worth highlighting since these two languages are not in the same language family, and the En-Zh dataset has a low-resource issue. The improvement also proves that my proposed method can effectively address both of the central issues in this field.

In Chapter 5, I further discussed the relationship between languages from the perspective of linearity and non-linearity. Given that linearity can offer competitive word translation result in languages that are in the same language family, and that non-linearity can provide competitive result in languages that are not in the same language family, I ask weather combining the two approaches together could provide a robust and overall a better result. Through my investigation in this topic, I made two contributions:

- I provide an ensemble-based method for cross-lingual word embeddings. I ensembled different linear and non-linear methods by combining their similarity matrices and rank order. I propose an ensemble-based cross-lingual word embedding method that achieves an improved result across all datasets utilised by Dinu and Baroni (2015).

- In the process of my work on this topic, I propose a new concept: Word consistency. By investigating the word consistency in different language pairs, I found that the some word pairs can be correctly translation through all experiments within word consistent pairs. I hypothesise that those word pairs could be used as new word pairs. I designed a consistency-based model which can effectively generate new word pairs from monolingual corpora. The generated word pairs could then be used for further learning high quality cross-lingual word embeddings.

I conducted numerous experiments on the consistency-based model. I found that newly generated word pairs can learn better cross-lingual word embeddings and provide a better result than individual models.

The focus of my research in Chapter 6 was speech translation. Current ST models suffer from data insufficiency. Multiple attempts exist to distil from rich-resource models to low-resource models like machine translation to speech translation model. Knowledge distillation methods are a popular way of transferring knowledge. However, in my investigation, I find that the teacher/student scenario does not fit MT and ST models. This chapter provides a new knowledge transfer scenario: mutual learning. Instead of one-

way learning from teacher to student, my proposed model can learn knowledge in both directions. Two contributions arise from this chapter. Firstly, I propose a mutual-learning scenario that simultaneously allows both MT and ST models to acquire knowledge. The proposed model can relax the searching space and provide better results on the MuST-C speech translation dataset. Secondly, The original mutual-learning model suffers from the KL vanishing issue. To mitigate this issue, the proposed model utilises the cycling annealing schedule. The experiments show that the proposed approach can successfully alleviate the KL vanishing issue.

There are numerous future research opportunities that this work has opened up, which I would like to continue investigating further. There are four topics:

- **The deep learning technologies for cross-lingual word embedding**: The focus of current studies has been to move from machine learning methods to deep learning methods. Pre-trained models like BERT or MUSE have offered competitive results in recent years. However, those models are costly to train or need tremendous data. Therefore, a light, efficient pre-train model for low-resource languages is would be beneficial. My future work will upgrade existing pre-trained transformer-based models by relaxing their requirement for extensive data and expensive resources.

- **Zero-shot cross-lingual word embeddings for extreme low-resource language or endangered languages.** Current advances for cross-lingual word embeddings provide optimism for translating endangered languages such as Maori or languages that have already become extinct, e.g. Tangut script. My future work will conclude the existing methods and link those languages with rich resource languages like English or French.

- **An end-to-end speech translation model for multi language speech translation.** A current advance in MT has been the extension from one-to-one machine translation to many-to-many translation. A single many-to-many model can translate languages effectively instead of training separate models for different language pairs. The many-to-many model can save training time and computational resources.

- **Zero-shot speech translated through meta-learning.** Meta-learning is the newest technology to deal with data insufficiency issues. Meta-learning provides impressive results in zero-shot image classification tasks and is now starting to be applied

in the NLP field. Meta-learning delivers the possibility of translating speech in low-resource languages, which would also be my subsequent research focus.

Current studies have tended to focus on rich-resource languages and supervised learning approaches. However, researchers have to face unlabelled, uncleaned data having a limited size in real-world applications. Therefore, most of my work has focused on unsupervised learning and mitigating data insufficiency issues. My overarching goal in this study is to tackle this complex challenge that is highly relevant to industry and real-world applications involving low-resource languages.

# Bibliography

Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. 2017. Cross-lingual word embeddings for low-resource language modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 937–947, Valencia, Spain. Association for Computational Linguistics.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A. Smith. 2016. Massively multilingual word embeddings. *CoRR*, abs/1602.01925.

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. 2013. Deep canonical correlation analysis. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, volume 28, pages 1247–1255, Atlanta, USA. Proceedings of Machine Learning Research.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 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 (EMNLP 2016)*, pages 2289–2294, Austin, USA. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 451–462, Vancouver, Canada. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018a. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2018)*, volume 32.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018b. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018c. Unsupervised neural machine translation. In *The 6th International Conference on Learning Representations (ICLR 2018)*, Vancouver, Canada.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *The 3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, USA.

Sameer Bansal, Herman Kamper, Adam Lopez, and Sharon Goldwater. 2017. Towards speech-to-text translation without speech recognition. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 474–479, Valencia, Spain. Association for Computational Linguistics.

Yehoshua Bar-Hillel. 1960. A demonstration of the non feasibility of fully automatic translation. *Advances in Computers (Appendix III of 'The present status of automatic translation of languages', Reprinted in Bar-Hillel Y, 1964)*, 1:158–163.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The journal of machine learning research (JMLR)*, 3:1137–1155.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, pages 6224–6228. IEEE.

Alexandre Berard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *CoRR*, abs/1612.01744.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings*

*of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21, Berlin, Germany. Association for Computational Linguistics.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2016. Quasi-recurrent neural networks. *CoRR*, abs/1611.01576.

Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning*.

Stéphane D'Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala. 2020. Double trouble in double descent: Bias and variance(s) in the lazy regime. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119, pages 2280–2290. Proceedings of Machine Learning Research.

Tijl De Bie and Bart De Moor. 2003. On the regularization of canonical correlation analysis. *Int. Sympos. ICA and BSS*, pages 785–790.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ben Dickson. 2020. The gpt-3 economy.

Georgiana Dinu and Marco Baroni. 2015. Improving zero-shot learning by mitigating the hubness problem. In *3rd International Conference on Learning Representations (ICLR 2015)*, San Diego, CA, USA.

Yerai Doval, Jose Camacho-Collados, Luis Espinosa-Anke, and Steven Schockaert. 2018. Improving cross-lingual word embeddings by meeting in the middle. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 294–304, Brussels, Belgium. Association for Computational Linguistics.

Long Duong, Antonios Anastasopoulos, David Chiang, Steven Bird, and Trevor Cohn. 2016. An attentional model for speech translation without transcription. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2016)*, pages 949–959, San Diego, California. Association for Computational Linguistics.

Tolga Ergen and Mert Pilanci. 2021. Global optimality beyond two layers: Training deep relu networks via convex programs. In *Proceedings of the 38th International Conference on Machine Learning (ICML 2021)*, volume 139, pages 2993–3003. Proceedings of Machine Learning Research.

Manaal Faruqui and 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 (EACL 2014)*, pages 462–471, Gothenburg, Sweden. Association for Computational Linguistics.

Mikel L Forcada, Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M Tyers. 2011. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144.

Hao Fu, Chunyuan Li, Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, and Lawrence Carin. 2019. Cyclical annealing schedule: A simple approach to mitigating KL vanishing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*, pages 240–250, Minneapolis, Minnesota. Association for Computational Linguistics.

Christian Fügen. 2009. *A system for simultaneous translation of lectures and speeches*. Ph.D. thesis, Karlsruhe Institute of Technology.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020a. End-to-end speech-translation with knowledge distillation: FBK@IWSLT2020. In *Proceedings*

*of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, pages 80–88, Online. Association for Computational Linguistics.

Marco Gaido, Mattia Antonino Di Gangi, Matteo Negri, and Marco Turchi. 2020b. On knowledge distillation for direct speech translation. *CoRR*, abs/2012.04964.

Di Gangi, Mattia A., Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019a. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2019b. Adapting Transformer to End-to-End Spoken Language Translation. In *INTERSPEECH 2019*, pages 1133–1137.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 123–135, Vancouver, Canada. Association for Computational Linguistics.

Samanwoy Ghosh-Dastidar and Hojjat Adeli. 2009. Spiking neural networks. *International journal of neural systems*, 19(04):295–308.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.

Qiushan Guo, Zhipeng Yu, Yichao Wu, Ding Liang, Haoyu Qin, and Junjie Yan. 2019. Dynamic recursive neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2019)*, pages 5147–5156.

Lars Kai Hansen and Peter Salamon. 1990. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 12(10):993–1001.

David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. 2004. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR 2016)*, pages 770–778.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. In *The 5th International Conference on Learning Representations (ICLR 2016)*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Frederick Jelinek. 1997. *Statistical methods for speech recognition*. MIT press.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aäron van den Oord, Alex Graves, and Koray Kavukcuoglu. 2016. Neural machine translation in linear time. *CoRR*, abs/1610.10099.

JT Kent, John Bibby, and KV Mardia. 1979. *Multivariate analysis*. Academic Press Amsterdam.

Tomáš Kočiský, Karl Moritz Hermann, and Phil Blunsom. 2014. Learning bilingual word representations by marginalizing alignments. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, pages 224–229, Baltimore, Maryland. Association for Computational Linguistics.

Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2003)*, pages 127–133.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of*

*the Association for Computational Linguistics (ACL 2018)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

S. Kullback and R. A. Leibler. 1951. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86.

Ryosuke Kuwabara, Jun Suzuki, and Hideki Nakayama. 2020. Single model ensemble using pseudo-tags and distinct vectors. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 3006–3013, Online. Association for Computational Linguistics.

Guillaume Lample, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *The 6th International Conference on Learning Representations (ICLR 2018)*.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10).

Bin Liu and Guosheng Yin. 2020. Chinese document classification with bi-directional convolutional language model. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020)*, pages 1785–1788.

Yuchen Liu, Hao Xiong, Jiajun Zhang, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019. End-to-End Speech Translation with Knowledge Distillation. In *INTERSPEECH 2019*, pages 1128–1132.

Zhiyuan Liu, Yankai Lin, and Maosong Sun. 2020. *Representation learning for natural language processing*. Springer Nature.

Ang Lu, Weiran Wang, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Deep multilingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 250–256, Denver, USA. Association for Computational Linguistics.

Lu Lu, Yeonjong Shin, Yanhui Su, and George Em Karniadakis. 2019. Dying relu and initialization: Theory and numerical examples. *CoRR*, abs/1903.06733.

Yehezkel Noa Lubin, Jacob Goldberger, and Yoav Goldberg. 2019. Aligning vector-spaces with noisy supervised lexicon. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2019)*, pages 460–465, Minneapolis, USA. Association for Computational Linguistics.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NeurIPS 2013)*, page 3111–3119, Red Hook, USA. Curran Associates Inc.

Tom M Mitchell et al. 1997. Machine learning.

Markus Müller, Thai Son Nguyen, Jan Niehues, Eunah Cho, Bastian Krüger, Thanh-Le Ha, Kevin Kilgour, Matthias Sperber, Mohammed Mediani, Sebastian Stüker, and Alex Waibel. 2016. Lecture translator - speech translation framework for simultaneous lecture translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations (NAACL 2016)*, pages 82–86, San Diego, USA. Association for Computational Linguistics.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin Dogus Cubuk, and Quoc V. Le. 2019. Specaugment: A simple augmentation method for automatic speech recognition. In *INTERSPEECH 2019*, pages 2613–2617.

Yves Peirsman and Sebastian Padó. 2010. Cross-lingual induction of selectional preferences with bilingual vector spaces. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational*

*Linguistics (NAACL-HLT 2010)*, pages 921–929, Los Angeles, California. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2018)*, pages 2227–2237, New Orleans, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. *CoRR*, abs/1804.08771.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2009. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009)*, page 865–872, New York, USA. Association for Computing Machinery.

Khalid Raza. 2019. Improving the prediction accuracy of heart disease with ensemble learning and majority voting rule. In *U-Healthcare Monitoring Systems*, pages 179–196. Elsevier.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Nicholas Ruiz and Marcello Federico. 2014. Assessing the impact of speech recognition errors on machine translation quality. In *Proceedings of the 11th Conference of the Association for Machine Translation in the Americas: MT Researchers Track*, pages 261–274, Vancouver, Canada. Association for Machine Translation in the Americas.

Ashutosh Saboo and Timo Baumann. 2019. Integration of dubbing constraints into machine translation. In *Proceedings of the 4th Conference on Machine Translation*, pages 94–101, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *The 5th International Conference on Learning Representations (ICLR 2017)*.

Matthias Sperber and Matthias Paulik. 2020. Speech translation and the end-to-end promise: Taking stock of where we are. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020)*, pages 7409–7421, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NeurIPS 2014)*, pages 3104–3112, Montreal, Canada.

Toshiyuki Takezawa, Tsuyoshi Morimoto, Yoshinori Sagisaka, Nick Campbell, Hitoshi Iida, Fumiaki Sugaya, Akio Yokoo, and Seiichi Yamamoto. 1998. A japanese-to-english speech translation system: Atr-matrix. In *5th International Conference on Spoken Language Processing (ICSLP 1998)*.

Graham Thurgood and Randy J LaPolla. 2016. *The sino-tibetan languages*. Taylor & Francis.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association.

Grigorios Tsoumakas and Ioannis Vlahavas. 2007. Random k-labelsets: An ensemble method for multilabel classification. In *18th European Conference on Machine Learning (ECML 2007)*, pages 406–417, Warsaw, Poland. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS 2017)*, pages 5998–6008, Long Beach, USA.

Laura Cross Vila, Carlos Escolano, José A. R. Fonollosa, and M. Costa-jussà. 2018. End-to-end speech translation with the transformer. In *IberSPEECH*, Barcelona, Spain. International Speech Communication Association.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *INTERSPEECH 2017*, pages 2625–2629.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2015)*, pages 1006–1011, Denver, USA. Association for Computational Linguistics.

Shuoheng Yang, Yuxin Wang, and Xiaowen Chu. 2020a. A survey of deep learning techniques for neural machine translation. *CoRR*, abs/2002.07526.

Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. 2020b. Rethinking bias-variance trade-off for generalization of neural networks. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119, pages 10767–10777. Proceedings of Machine Learning Research.

Pei Zhang, Niyu Ge, Boxing Chen, and Kai Fan. 2019. Lattice transformer for speech translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL 2019)*, pages 6475–6484, Florence, Italy. Association for Computational Linguistics.

Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. 2018. Deep mutual learning. In *Conference on Computer Vision and Pattern Recognition (CVPR 2018)*, pages 4320–4328, Salt Lake City, USA.

Haojie Zhao, Gang Yang, Wang Dong, and Lu Huchuan. 2021a. Deep mutual learning for visual object tracking. *Pattern Recognition*, 112:107796.

Jiawei Zhao and Andrew Gilman. 2020. Non-linearity in mapping based cross-lingual word embeddings. In *Proceedings of the 12th Language Resources and Evaluation Conference (LREC 2020)*, pages 3583–3589, Marseille, France. European Language Resources Association.

Jiawei Zhao, Wei Luo, Boxing Chen, and Andrew Gilman. 2021b. Mutual-learning improves end-to-end speech translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP 2021)*, pages 3989–3994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.