Analysis and Applications of Cross-Lingual Models in Natural Language Processing

by

Yoshinari Fujinuma

B.A., International Christian University, 2012

M.S., University of Tokyo, 2014

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2021

Committee Members:

Katharina Kann

Jordan Boyd-Graber

Michael J. Paul

Aaron Clauset

James H. Martin

Fujinuma, Yoshinari (Ph.D., Computer Science)

Analysis and Applications of Cross-Lingual Models in Natural Language Processing

Thesis directed by Prof. Katharina Kann

Human languages vary in terms of both typologically and data availability. A typical machine learning-based approach for natural language processing (NLP) requires training data from the language of interest. However, because machine learning-based approaches heavily rely on the amount of data available in each language, the quality of trained model languages without a large amount of data is poor. One way to overcome the lack of data in each language is to conduct cross-lingual transfer learning from resource-rich languages to resource-scarce languages. Cross-lingual word embeddings and multilingual contextualized embeddings are commonly used to conduct cross-lingual transfer learning. However, the lack of resources still makes it challenging to either evaluate or improve such models. This dissertation first proposes a graph-based method to overcome the lack of evaluation data in low-resource languages by focusing on the structure of cross-lingual word embeddings, further discussing approaches to improve cross-lingual transfer learning by using retrofitting methods and by focusing on a specific task. Finally, it provides an analysis of the effect of adding different languages when pretraining multilingual models.

Dedication

Dedicated to the memory of my grandmother Yoko Ito (1921-2017).

Acknowledgements

I would like to first sincerely thank all prior and current faculty members and students at University of Colorado Boulder. Out of all, especially thanking Jordan Boyd-Graber for providing me financial support and constructive advice throughout my PhD life to help me aim towards high-quality research regardless of being physically distant for most of the time, Katharina Kann for generously agreeing to advise me in my last year of PhD and giving me important advice and support on conducting high-quality research, and Michael J. Paul for opening up my path towards obtaining a PhD degree. I would like to also thank Aaron Clauset for agreeing to be part of my thesis committee member and providing initial insights on my first paper as a PhD student, James H. Martin and Martha Palmer for providing me advice and constructive feedback actively through the seminars, and finally Kenneth Anderson and Leysia Palen for offering me financial support through their grant. The members at the University of Colorado Boulder CLEAR lab, NALA lab, Paul lab, and from the EPIC project also provided me support and fruitful discussions. I especially like to thank Abteen Ebrahimi and Adam Wiemerslage for proofreading this dissertation.

Outside of University of Colorado, I would like to thank the following people: Hiroshi Suzuki, Alvin Grissom II, and Pascual Martínez-Gómez for advising me on surviving as a PhD student, Masato Hagiwara for detailed feedback on my code and writing, Komachi Mamoru and Akiko Aizawa for advising in the later stage of my PhD, Yogarshi Vyas and Miguel Ballesteros for acting as a mentor during my intern, members at the University of Maryland CLIP lab for providing feedback, Tsukasa Hibino for providing authentic Japanese food. Finally, thanking my family in Japan and my friends across the world for supporting me during my hard time.

Contents

Tables

Table

Figures

Figure

Chapter 1

Introduction

Not every person and place around the world communicates with the same language. According to Eberhard et al. (2021), there are 7139 living natural languages.[1] However, according to the International Association of Hyperpolyglots, there are only 160 people who are assessed as fluent in six or more languages (Chohan, 2021). Most people cannot learn every language perfectly, and therefore, a tool to bridge the gap among different languages all over the world is necessary. Bridging the gap among languages, such as a world where you can speak your first language and being able to communicate with the rest of the world, is one of the ultimate goals in the field of natural language processing.

Natural language processing (NLP) understands and manipulates human languages. After the introduction of machine learning-based approaches to NLP, data-driven approaches dominate NLP models. To name a few, word embeddings (Mikolov et al., 2013b) and contextual embeddings (Devlin et al., 2019) are the major milestones of recent advances in NLP. Development on such models primarily happens on languages used by many people, namely English. However, one of the goals of the NLP community is to promote language diversity and widen the scope of languages beyond such widely used languages.[2]

One way to bridge among languages is to have one model which works on many languages. Such multilingual models are beneficial because of the following reasons (Neubig, 2021): (1) it

[1] This sentence is inspired from the introduction section in Hao (2019).

[2] https://www.2022.aclweb.org/post/acl-2022-theme-track-language-diversity-from-low-resource-to-endangered-languages

enables cross-lingual transfer, where one trains a model in one language and apply it to another, and (2) it saves space on the computer storage when compared to having one model per 7000+ language, especially on recent large models. Ideally, multilingual NLP models are designed to work on *any* language. For example, if one is interested in a task to classify a given document into a specific topic, then having training data in any single language is sufficient to build a model that is applicable to any language. However, in real-world settings, these models do not always work and have certain conditions on *when* they would be successful or not. One such condition is the diversity of languages itself (e.g., English and Japanese are vastly different). Another condition is if the availability of linguistic resources differs across languages, then it works on languages with large resources available but not on languages where only a limited amount of resources are available. Finally, a model being successful or not depends on how it is trained prior to further tuning the model on the tasks and languages of interest. If a model is primarily designed and trained to work on English, it may work on other languages closely related to English such as Dutch, but may not work on languages like Japanese which have a different structure of language than English.

This dissertation analyzes and applies such multilingual models and reveal their limitations, and introduces an attempt to further improve them. The detailed structure of this dissertation is as follows.

Chapter 2 I explain the background and review related work on monolingual word embeddings, cross-lingual word embeddings, monolingual contextual embeddings, cross-lingual contextual embeddings, and the open challenges for them.

Chapter 3 I propose a method inspired from the network analysis literature to analyze and evaluate cross-lingual word embeddings by using a lexical graph derived from cross-lingual word embeddings. Specifically, I use a metric called graph modularity to quantify what will happen when cross-lingual word embeddings are not successful, for transfer between distant languages such as English and Japanese.

Chapter 4 I introduce our attempt to improve cross-lingual word embeddings inspired from the last chapter. Specifically, I combine graph-based approach to further improve cross-

lingual transfer learning. I further present an example application of cross-lingual embeddings in the context of readability assessment task and introduce a task-specific feature to further complement cross-lingual embeddings.

Chapter 5 I present our work on analyzing cross-lingual learning of pretrained multilingual models depending on different pretraining languages used. Specifically, I investigate the effect on languages where models have not been trained on such languages.

Chapter 6 I conclude and address future work on multilingual learning.

Chapter 2

Background

This chapter reviews the fundamental work on embeddings, which are used throughout this dissertation. I first review the monolingual word embedding method proposed by Mikolov et al. (2013b). Then, I review work on cross-lingual word embeddings which focus on learning a single projection matrix from one monolingual embedding space to another (Mikolov et al., 2013a). Finally, I cover the recent emerging trend in cross-lingual transfer learning which uses multilingual contextualized embeddings (Schuster et al., 2019; Devlin et al., 2019).

2.1 Overview

The idea of word embeddings is simple: assign a vector to represent the meaning of the word by predicting a word given its context, i.e., the surrounding words. Using the surrounding words is motivated by the distributional hypothesis (Harris, 1954), which is often phrased as "a word is characterized by the company it keeps" (Firth, 1957). By predicting a word given its surrounding words, words that are semantically similar to each other are placed close to each other in the embedding space. This helps machine learning-based NLP models to help generalize to words that do not appear in the training data.

Word embeddings (Bengio et al., 2003) is a common first step for multiple natural language processing tasks. One key work in word embeddings is the work by Mikolov et al. (2013b), which made it possible to obtain a word representation from a large unlabeled corpus (Section 2.2). Incorporating these pre-trained word representations has been shown to improve multiple downstream

tasks (Collobert and Weston, 2008).

Inspired by its success in the monolingual setting, there has been a rise of interest in cross-lingual settings. Machine learning-based NLP models rely on having a huge amount of training data. However, the quality and the amount of data available to train NLP models largely differ across languages. One approach to this problem is **cross-lingual transfer learning**, where I train NLP models using texts from different languages. **Cross-lingual word embeddings** is one approach in cross-lingual transfer learning, which enables exploiting large amounts of texts in resource-rich languages (e.g., English) and transferring them to resource-poor languages. As a result, it requires less or even none of the labeled data in the target languages to train a model for various NLP tasks. The work on cross-lingual word embeddings are discussed in depth in Section 2.3.

In the late 2010s, there is a shift of research interest from *static* cross-lingual word embeddings, where a single real-valued vector representation to each word, to *dynamic* or *contextualized* word embeddings (Jurafsky and Martin, 2020), where multiple real-valued vector representations to each word depending on its surrounding words. After its success in monolingual settings (Peters et al., 2018; Devlin et al., 2019), extending it to multilingual settings attracted interests from the NLP research community. These models are called **multilingual contextualized embeddings**, or more specifically, **multilingual language models**.[1] This is because, borrowing the example from (Schuster et al., 2019), contextualized embeddings enable English word "bear" to be close to multiple possible translations such as "oso" (animal) or "tener" (to have) depending on the context. Such multilingual contextualized embeddings bring consistent improvement on multiple NLP tasks over multiple languages, for example on parsing (Ahmad et al., 2019b) and document classification (Eisenschlos et al., 2019), when compared to static cross-lingual word embeddings. The work on multilingual contextualized embeddings are discussed in depth in Section 2.4.

[1] Since these models are not explicitly designed to make it aligned across languages, I call it "multilingual" rather than "cross-lingual" throughout this dissertation.

2.2 Monolingual Word Embeddings

More technically, word embedding methods map each word into a dense d-dimensional continuous vector (which d is typically set within the range of $50 - 1000$) given an unlabeled monolingual corpus. Though there are various ways to do so in the NLP literature (e.g., latent semantic analysis (Deerwester et al., 1990), or latent Dirichlet allocation (Blei et al., 2003)), I focus on skip-gram with negative sampling (Mikolov et al., 2013b, SGNS) in this dissertation because of its in-depth analysis (Levy and Goldberg, 2014; Mimno and Thompson, 2017), and its extension to exploit subword-level similarities (Bojanowski et al., 2017), which is commonly used as inputs to the cross-lingual word embedding methods (Section 2.3).

I now review SGNS in-depth. SGNS produces a distributional word vector representation motivated by the distributional hypothesis (Harris, 1954). While training SGNS, it keeps track of two different vectors associated with each word i in the vocabulary: (1) word vector $w_i \in \mathbb{R}^d$, and (2) context vector $c_i \in \mathbb{R}^d$. The first step of SGNS is to extract context words given a word i by extracting the surrounding words of the word i that appear within a given window size as *positive context*. Then, sample S words as *negative context*, which is assumed to not appear as surrounding context words for i. Given these positive and negative context words, SGNS maximizes the dot product of a word vector w_i and positive context vectors c_j, and minimizes the dot product of the word vector w_i and negative context vectors c_j for $j = 1, ..., S$ by optimizing the following loss function L_{SGNS}:

$$L_{\text{SGNS}}(w_i, c_j) = -\log \sigma(w_i^T c_j) - \sum_{s}^{S} \log \sigma(-w_i^T c_s), \tag{2.1}$$

where σ is the sigmoid function. The resulting word vector w_i are the word embedding for word i and context vectors c_j are either discarded or combined with word vectors to obtain final word embeddings (Pennington et al., 2014). One important indication of Equation 2.1 is that it optimizes the dot product, i.e., $w_i^T c_j = \|w_i\|\|c_j\|\cos(w_i, c_j)$, which also encodes information to the norm of word vectors $\|w_i\|$ (e.g., word frequency of an input monolingual corpus (Arora et al., 2016)), which could vary across different monolingual embeddings. Thus, preprocessing word vectors to make it

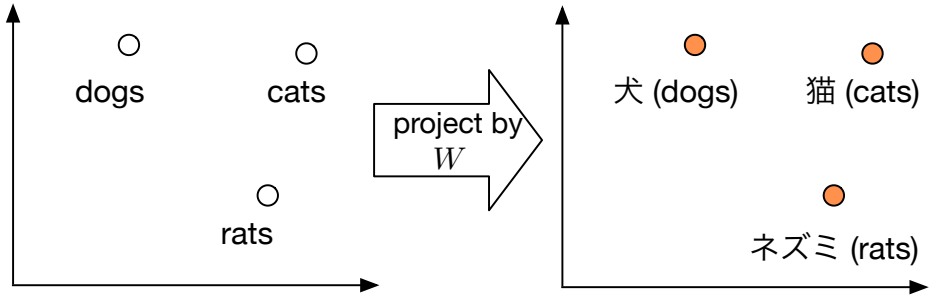


Figure 2.1: A toy figure inspired by Mikolov et al. (2013a) which shows the similar geometric arrangement between two monolingual embeddings in English (left) and Japanese (right). This similarity in geometric arrangement motivates to obtain a cross-lingual word embeddings by projecting one embedding space to another using a linear projection matrix W.

unit length i.e., $\|w_i\| = 1$ also mentioned in Section 2.3.1.

Equation 2.1 has also been further extended to exploit subword-level similarities. This is accomplished by making use of an n-gram, which is a sequence of n given linguistic unit (which are typically words or characters). Specifically, Bojanowski et al. (2017) proposed a model called fastText to keep track of separate vectors for character n-grams while training SGNS. The pretrained fastText vectors in 50+ languages on Wikipedia in each language provided by Bojanowski et al. (2017) has been a popular choice of pre-trained word embeddings in multiple languages.

In the next section, I introduce approaches to extend monolingual word embeddings into cross-lingual settings.

2.3 Cross-Lingual Word Embeddings

This section discusses projection-based cross-lingual embedding initially proposed by Mikolov et al. (2013a). Mikolov et al. (2013a) observe that among the pre-trained monolingual embeddings, words that have similar meanings across languages had similar geometric arrangements (Figure 2.1). This observation motivated them to train a cross-lingual linear projection $W \in \mathbb{R}^{d \times d}$ to map word vectors from one monolingual embedding space to another.

A cross-lingual projection W is used to align the two embeddings by projecting vectors from one embedding space to another. Training such projection W is as follows. Given a bilingual

lexicon (e.g., extracted from Panlex[2] or Wiktionary[3]), and pre-trained monolingual embeddings in two different languages, the first step is to build a stack of a aligned vectors $X \in \mathbb{R}^{a \times d}$ and $Y \in \mathbb{R}^{a \times d}$ from source and target language embedding space. Then, a cross-lingual projection matrix W minimizes the loss function L of the distance of aligned vectors from each language i.e.,

$$L(W) = \sum_{j=1}^{d} \sum_{i=1}^{a} ((Wx_i)_j - (y_i)_j)^2 = \sum_{i=1}^{a} \|Wx_i - y_i\|_2^2 = \|WX - Y\|_F^2 \tag{2.2}$$

where F is the Frobenius norm, and $W \in \mathbb{R}^{d \times d}$ is the cross-lingual projection from the source embedding space to the target embedding space. The projection-based cross-lingual embedding methods discussed in this dissertation either extends or incorporate Equation 2.2 as one step in the training pipeline to obtain cross-lingual word embeddings.

2.3.1 Orthogonal Projections

One problem of cross-lingual word embeddings is that the similarity of word vectors can be different from the original embedding space. For example, if the similarities of the vectors of the semantically similar English words "dog" and "cat" were high in the original English embedding space, we want this to be preserved in the projected cross-lingual word embedding space too. Preserving the similarity of the word vectors in the original embedding space can be realized through what is called the orthogonal matrix.

Xing et al. (2015) use the orthogonal constraint on a cross-lingual projection W to improve the cross-lingual embeddings, which has also been shown to work empirically by others (e.g., Artetxe et al. (2016)). As mentioned by Xing et al. (2015), a cross-lingual projection W has a risk of changing the intra-lingual similarity of word vectors, measured by dot products or cosine similarities, in the original monolingual embedding space. One way to preserve the similarities of vectors, namely the dot product, is to add the orthogonal constraint $W^T W = I$ where I represents the identity matrix. This is because by applying a projection W to a given vector x and taking the dot product

[2] `https://panlex.org/`
[3] `https://www.wiktionary.org/`

of the projected vector Wx becomes

$$(Wx)^T Wx = x^T W^T Wx = x^T x \tag{2.3}$$

because of the orthogonal constraint $W^T W = I$. Note that by further preprocessing the input vector x be of unit length i.e., $\|x\|_2 = 1$, the dot product $x^T x$ and the cosine similarity $\cos(x, x)$ become equivalent since $x^T x = \|x\|_2 \|x\|_2 \cos(x, x) = \cos(x, x)$. Adding orthogonal constraints to Equation 2.2 leads to the well-known orthogonal Procrustes problem (Schönemann, 1966), i.e.,

$$L(W) = \|WX - Y\|_F^2 \text{ s.t. } W^T W = I \tag{2.4}$$

where $I \in \mathbb{R}^{d \times d}$ is the identity matrix and F is the Frobenius norm.

The orthogonal constraint has been empirically shown to work well both for supervised, where a bilingual lexicon is used as a supervision signal, and unsupervised cross-lingual embedding methods, where a bilingual lexicon is not used as a supervision signal. In addition, the orthogonal constraint has two more benefits: (1) Though the orthogonal Procrustes problem is non-convex, one can obtain a globally optimal W by using the singular value decomposition (Schönemann, 1966, SVD) by taking $W = UV^T$ such that $U\Sigma V^T = \text{SVD}(YX^T)$, where Σ are the singular values, and (2) it avoids overfitting W to the translation pairs in the given bilingual seed lexicon.

2.3.2 Supervised Cross-Lingual Word Embeddings

This section discusses how to obtain cross-lingual word embeddings using a bilingual lexicon as the supervision signal. The projection-based cross-lingual embedding methods that use some external human-labeled resources (e.g., bilingual lexicons) are called **supervised** and methods without using human-labeled resources are called **unsupervised**. Since Mikolov et al. (2013a) observe the geometric correlation between translations across monolingual embeddings from different languages, many researchers propose a linear projection-based method to preserve its intra-lingual relationship after projecting it to target language embedding space.

One line of research in supervised cross-lingual word embeddings is to project both source and target word vectors to a shared cross-lingual space that is neither the source nor the target em-

bedding space. Though many methods are based upon solving the orthogonal Procrustes problem to obtain the linear cross-lingual projection W, there are other approaches. For example, Faruqui and Dyer (2014) propose a canonical correlation-based approach. Specifically, given aligned vector pairs from two languages, they train linear projections to a shared embedding space that maximizes the correlation between the given aligned vector pairs.

Another line of research on supervised cross-lingual embedding methods is to use as small a bilingual lexicon as possible to minimize human effort. Artetxe et al. (2017) report that by iteratively applying the orthogonal Procrustes problem on the newly induced bilingual seed lexicon, it only requires a small initial number of translation pairs in a bilingual lexicon (e.g., 25 pairs) to obtain a good cross-lingual word embedding. This approach is later used as the "Refinement Step" to other methods including the unsupervised methods proposed by Conneau et al. (2018a) and Hoshen and Wolf (2018). The new seed bilingual lexicon obtained at every training step uses similarity metrics such as the top similar dot product (Artetxe et al., 2017) or cross-domain similarity local scaling- (Conneau et al., 2018a, CSLS), which we further explain in Section 2.3.4.

Finally, Relaxed CSLS (Joulin et al., 2018, RCSLS) uses CSLS (Conneau et al., 2018a) directly as the loss function. Assuming all input vectors are unit length normalized, importing CSLS to the loss function results in the following loss function L:

$$L(W) = \frac{1}{n}\sum_{i=1}^{n} -2x_i^T W^T y_i + \frac{1}{k}\sum_{y_j \in N_Y(Wx_i)} x_i^T W^T y_j + \frac{1}{k}\sum_{Wx_j \in N_X(y_i)} x_j^T W^T y_i \tag{2.5}$$

where $N_X(y_i)$ is the k-nearest neighbors of the vector y_i from the target embedding Y in the source embedding space X.

Joulin et al. (2018) further add the convex-relaxed version of the orthogonal constraint to Equation 2.5. Remind that the spectral norm $\|W\|_{op}$ (i.e., operator 2 norm) is equivalent to the maximum singular value of W. The convex-relaxed version of the orthogonal constraint is constraining W to be on the unit spectral norm ball $\|W\|_{op} < 1$. Joulin et al. (2018) compare the accuracy on the bilingual lexical induction task using Equation 2.5 with and without this constraint $\|W\|_{op} < 1$, and they report that the one without this constraint scored higher accuracy than the

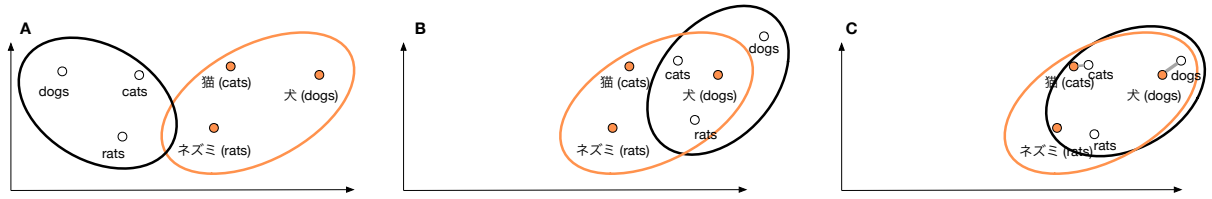


Figure 2.2: A toy figure of MUSE inspired from the image by Conneau et al. (2018a). **(A)** shows the two monolingual embeddings (English in black, and Japanese in orange) in different languages, **(B)** shows that the two monolingual embeddings are roughly aligned after the adversarial step, and **(C)** shows the refinement step using cross-lingual word vectors that are close to each other from Step **(B)**.

constrained one on the MUSE benchmark test set (Conneau et al., 2018a). This result further re-opens the research direction on cross-lingual projections without orthogonal constraints.

2.3.3 Unsupervised Cross-Lingual Word Embeddings

So far, we have assumed that we are given a bilingual lexicon to train a cross-lingual projection matrix W. However, obtaining the seed bilingual lexicon is either not possible or can be expensive for low-resource languages. This motivated many researchers to focus on finding a way to train cross-lingual embeddings without a bilingual seed lexicon. Though there are various methods proposed recently (Zhang et al., 2017; Artetxe et al., 2018; Hoshen and Wolf, 2018; Alvarez-Melis and Jaakkola, 2018; Xu et al., 2018; Chen and Cardie, 2018), here, we introduce MUSE (Conneau et al., 2018a), which is a popular choice to embed words from multiple languages and use those for downstream NLP tasks, e.g., machine translation (Lample et al., 2018; Gu et al., 2018).

2.3.3.1 MUSE by Conneau et al. (2018a)

We now describe MUSE in depth. MUSE consists of two steps: (1) the adversarial step (**B** in Figure 2.2), and (2) the refinement step (**C** in Figure 2.2). Within the adversarial step, it also has two sub-steps: (a) Training a cross-lingual projection matrix W, and (b) training a language classifier D.

Adversarial Step In this step, we alternately train the language classifier D and the cross-lingual mapping W to make the language classifier hard to discriminate the language of a given word vector.

The language classifier D predicts the language $\hat{y} \in \{\text{source language}, \text{target language}\}$ of a word i given its vector x_i by $\hat{y}_i = D(x_i)$. D is a multi-layer perceptron with $h \in \mathbb{N}$ hidden units, where the output $z_l \in \mathbb{R}^h$ from each layer l is calculated by

$$z_l = \text{LeakyReLU}(W_l z_{l-1} + b_l) \tag{2.6}$$

where $W_l \in \mathbb{R}^{h \times h}$ is the weight, and $b_l \in \mathbb{R}$ is the bias of each layer l

$$\text{LeakyReLU}(x) = \begin{cases} x & (x \geq 0) \\ ax & (x < 0) \end{cases} \tag{2.7}$$

and $a \in \mathbb{R}$ is a constant hyperparameter. The final layer is defined as

$$\hat{y} = \sigma(W_L z_{L-1} + b_L) \tag{2.8}$$

where σ is the sigmoid function, $W_L \in \mathbb{R}^{h \times 1}$ is the weight, and $b_L \in \mathbb{R}$ is the bias of the final layer L.

The overall loss function L_{MUSE} is defined as the followings: for learning a cross-lingual mapping W,

$$L_{\text{MUSE}}(W) = -\frac{1}{n} \sum_{i}^{n} \log D(W x_i) \tag{2.9}$$

and for learning a language classifier D,

$$L_{\text{MUSE}}(D) = -\frac{1}{n} \sum_{i}^{n} \log D(W x_i) - \frac{1}{m} \sum_{j}^{m} \log D(y_j). \tag{2.10}$$

This adversarial step is inspired from an adversarial approach based on reversing the gradients during the back-propagation (Ganin and Lempitsky, 2015). In addition, Conneau et al. (2018a) use the Perseval tightness (Cisse et al., 2017)

$$\frac{\beta}{2} \left\| W^T W - I \right\|_2^2 \tag{2.11}$$

where $\beta \in \mathbb{R}$ is the constant hyperparameter, to keep the projection matrix $W \in \mathbb{R}^{d \times d}$ approximately orthogonal.

Refinement Step After obtaining a bilingual seed lexicon induced from the adversarial step, it iteratively applies the Procrustes approach, which is shown to be effective by Artetxe et al. (2017).

Validation Metric Since MUSE is unstable throughout training, it uses a validation metric to select the best cross-lingual projection matrix W. Conneau et al. (2018a) use CSLS, which we explain in the next section, of the top 10K frequent words, which is shown to be roughly correlated to the accuracy on the bilingual lexicon induction task, as the validation metric.

2.3.4 Evaluating Cross-Lingual Word Embeddings

With few exceptions (e.g., Ammar et al. (2016)), most papers on cross-lingual word embedding methods evaluate on the **bilingual lexicon induction** task. The bilingual lexicon induction task (or equivalently, the word translation task) is that given a word in a source language, I retrieve the correct translation in a target language. Though there are other tasks used to evaluate cross-lingual word embeddings (e.g., cross-lingual document classification (Klementiev et al., 2012) or syntactic parsing (Ammar et al., 2016)), those tasks are on either document- or sentence-level, which is stated as sub-optimal for evaluating cross-lingual word embeddings (Ruder et al., 2017). Intrinsic evaluation measures are also proposed to evaluate cross-lingual word embeddings, such as QVEC-CCA (Ammar et al., 2016), CSLS (Conneau et al., 2018a), and the eigenvalue similarity (Søgaard et al., 2018). Søgaard et al. (2018) show that the eigenvalue similarity between two monolingual lexical graphs derived from monolingual embeddings is correlated with the precision@k on bilingual lexical induction task.

QVEC and QVEC-CCA QVEC (Tsvetkov et al., 2015) is a score indicating the quality of learned word embeddings given oracle word embeddings computed from linguistic resources. It is calculated as the sum of the maximum possible Pearson's correlation $\rho \in [-1, 1]$ of one specific dimension of two different embeddings. Formally, given $n \in \mathbb{N}$ words and its stacked pre-trained

word embedding matrix $X \in \mathbb{R}^{d \times n}$ and a oracle word embedding matrix $S \in \mathbb{R}^{p \times n}$ (e.g., frequency of each sense of words computed from a sense-tagged corpora with $p \in \mathbb{N}$ senses),

$$\text{QVEC}(X, S) = \max_{\sum_j a_{ij} \leq 1} \sum_i \sum_j \rho(X_i, S_j) a_{ij} \tag{2.12}$$

where i and j are one dimension of each embedding matrix, and $a_{ij} \in \{0, 1\}$ is a binary value which represents whether dimension i in X and dimension j in S are aligned or not. $\sum_j a_{ij} \leq 1$ guarantees that one dimension from the pre-trained word embedding is aligned to at most one dimension from oral word embedding (Tsvetkov et al., 2015).

The downside of QVEC scores are that they are not comparable between word embeddings that have different dimensions. Since QVEC is the sum of Pearson's correlation on all dimensions, it is proportional to the embedding dimension d, and therefore, for example, embeddings with dimension size 200 tend to have higher QVEC score than the dimension size 50.

To have a metric which is comparable across different dimension size, Tsvetkov et al. (2016) extend QVEC to use canonical correlation (QVEC-CCA) defined as

$$\text{QVEC-CCA}(X, S) = \max_{v,u} \rho(X^T v, S^T u), \tag{2.13}$$

where v and u are learned vectors to maximize correlations between the two embedding matrices. The output of QVEC-CCA is a correlation within the range of $[-1, 1]$ regardless of the embedding dimension size.

Cross-domain Similarity Local Scaling (CSLS) Cross-domain Similarity Local Scaling (Conneau et al., 2018a, CSLS) is a cross-lingual similarity metric that penalizes words that appear in a dense region to ameliorate the hubness problem, which is a problem that a vector in a high-dimensional space often encounters the problem of "hubs" (Radovanović et al., 2010; Dinu et al., 2014) or a vector that frequently appears as nearest neighbors of arbitrary vectors. The hub vectors in cross-lingual embedding space are problematic when retrieving the translation of a given source word in the bilingual lexicon induction task. Though there are other methods to mitigate the hubness problem in the word embedding space (Smith et al., 2017), I explain in-depth the one proposed as a validation metric by Conneau et al. (2018a).

CSLS is defined as

$$\textsc{csls}(Wx, y) = 2\cos(Wx, y) - r(Wx) - r(y) \tag{2.14}$$

where $r(x)$ is the average cosine similarity of the top k cross-lingual nearest neighbors of a word vector x (k is usually set to 10), and W is the cross-lingual projection matrix. This cross-lingual similarity metric has become the de facto choice for cross-lingual nearest neighbor retrieval for the bilingual lexicon induction task. Moreover, this metric has been later incorporated into the loss function by Joulin et al. (2018) and achieved improved results on the MUSE benchmark dataset (Conneau et al., 2018a).

2.4 Multilingual Contextualized Embeddings

One major downside of using static word embeddings is handling words that have multiple senses (e.g., the river "bank" vs. a financial establishment "bank") since a given word is always mapped to the same embedding or a vector in any context or surrounding words. As opposed to "static" word embeddings, an emerging line of research is on "contextualized" embeddings. Contextualized embeddings refers to models that embed tokens (which can be words, subwords, characters, or even bytes) dynamically depending on the context or surrounding words. In this dissertation, I focus on contextualized embeddings obtained by a pretrained language model (Peters et al., 2018; Devlin et al., 2019). I first introduce the monolingual models in Section 2.4.1, followed by introducing its multilingual extensions in Section 2.4.2.

2.4.1 Monolingual Contextualized Embeddings

When training a model using contextualized embeddings, a common practice is to follow a **pretraining–finetuning paradigm** (Hinton et al., 2006), which has been first successfully applied to train machine learning models on images, and later imported to train models for texts. To train neural networks that have more layers and larger embedding dimensions, the training data from the task of interest is often scarce and not enough to train a good neural network model. Given a

task, the first step to train a model is called the **pretraining step** where the model parameters are trained on a given pretraining task, which is different from the task of interest. The second step is called **fine-tuning step** which uses the model pretrained from the first step and further trains it on the target task of interest. [4]

Early work on contextualized embeddings explore different pretraining tasks. McCann et al. (2017) use machine translation as the pretraining task and successfully fine-tuned and applied the pretrained model to multiple other tasks. However, one problem of such an approach is that the machine translation task uses parallel corpus as inputs to train a model, where the size of available corpus can be limited compared to a raw corpus. Later, Peters et al. (2018) show that pretraining using a stacked Long Short-Term Memory (Hochreiter and Schmidhuber, 1997, LSTM) on a large raw corpus using language modeling as the pretraining task and then successfully fine-tuned it on multiple downstream tasks. However, most recent work uses the model proposed by Devlin et al. (2019) or its variants, which uses the architecture called Transformers (Vaswani et al., 2017) and using the pretraining task called "masked language modeling" which I explain in the following section.

2.4.1.1 Transformers

Similar to static word embeddings, Transformers (Vaswani et al., 2017) also map input texts into continuous vector representations. Unlike LSTM, Transformers do not process one token after another and rather attend on surrounding tokens that are important for a given token.

Given a sequence of N tokens, the sequential operation takes $O(N)$ for recurrent neural networks such as LSTM, but $O(1)$ for Transformers due to not processing the inputs recurrently one token after another. This nature of fast computation with respect to the input sequence length resulted in pretraining on a larger scale, namely with more model parameters and larger corpora.

I now introduce the typical inputs to Transformers in the field of NLP. Such typical inputs

[4] There are recent work on adding an intermediate step (e.g., Pruksachatkun et al. (2020)) to further improve the model, but I will leave this as out of the scope for this dissertation.

to Transformers are: (1) positional embeddings, and (2) token embeddings. A variant of the Transformer model widely used in NLP (Devlin et al., 2019) takes an additional input called (3) segment embeddings. I assume that $d_m \in \mathbb{N}$ is the hyperparameter representing the dimension of a Transformer model.

Positional Embeddings Positional embeddings $p_i \in \mathbb{R}^{d_m}$ are trainable real-valued vectors associated with each position i of the input sequence. One main difference between the models prior to Transformers (e.g., LSTM) is that it no longer processes the input from left-to-right one token after another. This enables fast processing of input texts as a whole, but it loses the position information of a given token. Position information is important because multiple natural language processing tasks use the position information in a given sentence. For example, the following two sentences "The dog chases the cat" and "The cat chases the dog" use the same words, but they mean something completely different which can only be distinguished by the position of each word. To keep track of such information for Transformers, positional embeddings are one of the inputs.

Token Embeddings Token embeddings $t_i \in \mathbb{R}^{d_m}$ are trainable vectors which are technically the same as the static word embeddings. When given an input string, the input string is tokenized into tokens using a given vocabulary, and then further embedded into a vector using token embeddings which are associated with each string in the vocabulary. However, one difference is that as indicated by using the term "token embeddings" instead of "word embeddings", the input is no longer assumed to be a word.

Segment Embeddings Segment embeddings $s_i \in \mathbb{R}^{d_m}$ are trainable vectors embedded from binary representations associated to each input token to keep track of whether a given token is from the first sentence or the second sentence. For example, if a given sentence pair where the first sentence is "I like cats ." and the second sentence is "Cats like dogs .", then the inputs to compute segment embeddings are $\{0, 0, 0, 0, 1, 1, 1, 1\}$. For models which do not use the next sentence prediction task during pretraining, these segment embeddings are omitted.

For each token at position i in a given sequence, these input vectors are simply summed up into one input vector ($x_i = p_i + s_i + t_i$) and passed into Transformers.

At a high level, one key component in Transformers is the self-attention mechanism (Lin et al., 2017).[5] Self-attention allows to attend or assign different weights to the context or surrounding tokens within the same input sequence. To compute the self-attention, Transformers projects a given input vector x_i to three different vectors called **query**, **key**, and **value** using three different trainable parameters associated to each kind of vector i.e., W_j^Q for queries, W_j^K for keys, and W_j^V for values. The output of the attention function is the weighted sum of the value vectors V. The weights to compute such weighted sum are computed based on a query vector $q \in \mathbb{R}^{d_m}$ and a key vector $k \in \mathbb{R}^{d_m}$.

I now explain the architecture of Transformers in depth. Given a query matrix $Q \in \mathbb{R}^{N \times d_m}$, a key matrix $K \in \mathbb{R}^{N \times d_m}$, and a value matrix $V \in \mathbb{R}^{N \times d_m}$, self-attention is given by

$$\text{Attention}(Q, K, V) = \text{Softmax}(QK^T \odot \frac{1}{\sqrt{d_K}})V \tag{2.15}$$

where $\odot$ is the Hadamard product (i.e., element-wise multiplication). Most models have multiple attention modules or "heads", which are referred to as multi-head attention. In multi-head attention, each head computes different weights from the same input sequence. The computed attention output vectors are later concatenated and then applying a projection matrix W, i.e.,

$$\text{Multihead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h)W \tag{2.16}$$

where h is the number of attention heads and

$$\text{head}_j = \text{Attention}(QW_j^Q, KW_j^K, VW_j^V) \tag{2.17}$$

where W_j^Q, W_j^K, W_j^V are the projection matrices for Q, K, V, and W is the projection matrix applied to the concatenated output of the attention heads. Due to the matrix multiplication in each attention head, the computation complexity of each head_i is $O(N^2)$, which makes it harder to scale to the long sequences.

[5] There is also recent work by Tay et al. (2021a) that questions whether the self-attention mechanism is essential or not but I will keep it out of scope in this dissertation.

2.4.1.2 Major Architectures of Contextualized Embeddings

I briefly introduce three widely-used architectures for contextualized embeddings.

ELMo (Peters et al., 2018) ELMo is a three-layer bi-directional LSTM model, which is trained on the language modeling task, i.e., predicts the probability of the next word given context words. The inputs to the first layer of ELMo are characters of each token. ELMo has shown to improve multiple downstream NLP tasks by large margins.

BERT (Devlin et al., 2019) BERT is a variant of a Transformer model (Vaswani et al., 2017) which takes bi-directional inputs (i.e., both from left and right context tokens) trained on two unsupervised tasks: (1) a token- or a subword-level task called "masked language modeling" (MLM), and (2) a sentence-level task called "next sentence prediction". Masked language modeling is a task to predict randomly masked words from the surrounding words on a given percentage (another hyperparameter typically set to 15%) of all words in a corpus. For example, given the sentence "Cats [MASK] dogs", the model trained on masked language modeling predicts the words that should go in the [MASK] token, e.g., "chase".

Next sentence prediction is a task that is given an input sentence and a candidate sentence, where 50% of the time the candidate sentence is replaced by a random sentence from the training corpus, the model predicts whether the candidate sentence is the next sentence of the input sentence or not. Since the inputs to the model are now two sentences, BERT uses the segment embeddings as the third input embedding in addition to positional and token embeddings. The inputs to the first layer of BERT are subword-level tokens segmented by WordPiece (Wu et al., 2016). It has shown to improve multiple downstream NLP tasks by some margin over ELMo. The publicly available pretrained models[6] use BooksCorpus dataset and Wikipedia (16GB in total) to pretrain on it (Liu et al., 2020).

RoBERTa (Liu et al., 2020) BERT has been further improved by a model called RoBERTa. Referring from Liu et al. (2020), the key elements of RoBERTa compared to BERT are (1) training on larger data such as 16GB of BooksCorpus dataset (Zhu et al., 2015), 76GB

[6] `https://github.com/google-research/bert`

of Common Crawls News dataset (Nagel, 2016), 38GB of OpenWebText dataset (Gokaslan and Cohen, 2019), and 31GB of Stories dataset (Trinh and Le, 2019), (2) omitting the next sentence prediction task during pretraining, and only using masked language modeling, (3) longer input sequence, larger batch, and longer training steps, and (4) dynamic masking while pretraining using masked language modeling, or masking random words for every sequence right before inputting to the model rather than conduct masking as the preprocessing step.

BERT, RoBERTa, or its variants are dominant in the current NLP research because of its scalability to training it on large text data, and pretraining happens on the whole model rather than only at the embedding layer. Next section describes multilingual contextual embedding models that are trained on multiple languages by a language modeling task, which surprisingly work well compared to previous best practices on certain languages regardless of not using any bilingual resources.

2.4.2 Multilingual Pretrained Language Models

Early work on the multilingual language models use explicit cross-lingual supervision signals such as bilingual dictionaries (Schuster et al., 2019) or parallel corpora (Aldarmaki and Diab, 2019). Later, multilingual BERT (Devlin et al., 2019, mBERT) jointly trains a single BERT model on 100+ languages without using any cross-lingual supervision signals such as a parallel corpus or a bilingual dictionary. Multilingual pretrained language models are reported to transfer and scale across multiple languages surprising well even though it does not use any cross-lingual supervision signals (Pires et al., 2019). Recently, Wu et al. (2020) reported that monolingual BERT models has also been reported have structural similarity across languages, which is similar to the static word embedding methods having similar geometric arrangements (Mikolov et al., 2013a).

I now list and describe widely used multilingual language models. I also summarize the major similarities and differences among such pretrained multilingual language models in Table 2.1. There are more recent multilingual models in the literature, for example mT5 (Xue et al., 2021b), but I will focus on the following multilingual models in this dissertation.

Model	Pretraining Corpus		Pretraining Task		Vocab. Size	# Langs
	Wikipedia	Common Crawl	NSP	MLM		
mBERT (cased)	✓		✓	✓	119,547	104
XLM-17	✓			✓	200,000	17
XLM-100	✓			✓	200,000	100
XLM-R (base)	✓	✓		✓	250,000	100

Table 2.1: Main differences across pretrained multilingual language models explored in this dissertation. The models are classified by (1) the pretraining corpus used, (2) the pretraining task used such as next sentence prediction (NSP) and masked language modeling (MLM), (3) the vocabulary size, and (4) the number of languages used during pretraining.

Multilingual BERT (Devlin et al., 2019, mBERT) mBERT is a single BERT model trained on 100+ languages without using any cross-lingual supervision except shared vocabularies across languages. Since this model is the first publicly released multilingual language model that is trained on 100+ languages, it has been analyzed and studied in depth in the literature.

XLM (Lample and Conneau, 2019) XLM further adds two additional elements to BERT, where one is adding the fourth additional input called language embedding or a trainable vector associated to each language, and the other is using additional pretraining task called Translation Language Modeling (TLM). However, I only compare against XLM models pretrained without language embeddings and only using masked language modeling due to (1) having models pretrained on various numbers of languages, and (2) to be comparable with other models. Namely, I use XLM-17 and XLM-100.

XLM-RoBERTa (Conneau et al., 2020, XLM-R) XLM-R is a RoBERTa model pretrained on Common Crawls and Wikipedia corpus across 100 languages. Compared to mBERT, it has a larger vocabulary size, uses SentencePiece (Kudo and Richardson, 2018) instead of WordPiece (Schuster and Nakajima, 2012) for tokenizing input strings, and larger and more corpora are used for pretraining.

2.5 Language Similarity

One common issue in cross-lingual transfer learning is that the downstream stream task results largely depend on 1) the choice of both source languages to train the model on, and 2)

the target language to evaluate the model on. For analysis and practical purpose, computing the similarities across languages is an important topic in cross-lingual NLP. How can we measure the similarities between languages? Commonly used measure in the literature are either based on the language families, which are often represented by a phylogenetic tree (e.g., Hammarström et al. (2021)), syntactic structures (e.g., noun-adjective order), or dominant word order (e.g., SOV, SVO). All such features are accessible through language typology databases, such as World Atlas of Language Structures (Dryer, 2013, WALS) or Ethnologue (Eberhard et al., 2021).

In practice, all such language typological features from multiple databases are easily accessible by `lang2vec` (Littell et al., 2017), which is a Python library that maps each language to a single vector using multiple language typological databases. Lauscher et al. (2020) use WALS and other databases through `lang2vec` and show that the language similarities based on typological features (syntactical, phonetical, and grammatical) have high correlation between various downstream task results, such as POS tagging, named entity recognition, and natural language inference, using a fine-tuned multilingual pretrained langauge models.

In addition to typological features, language similarities can be empirically calculated from texts in each language. For example, Ahmad et al. (2019a) calculated the similarities among languages based on the sentence structure (the left or right direction of the dependency) by focusing specifically on dependency parsing task. Later, Lin et al. (2019) conduct detailed study on the selection of transfer language based on data-dependent and data-independent features, where data-dependent features are the empirical language similarities calculated from texts, and data-independent features (i.e., language typological features).

2.6 Challenges in Cross-Lingual Learning

I now mention some open problems and difficulty in learning a cross-lingual model.

Linguistic Units of a Token One open question for cross-lingual models is what linguistic units a model should consider. If focusing to consider words as tokens, then the issue of the tokenization arises for languages without explicit whitespaces between words (e.g., Chinese,

Japanese, Korean, and Thai), and also languages that use compound words as part of the grammar rule (e.g., German). Word segmentation or tokenization is a fundamental step in NLP, which has been researched in Japanese (Kudo et al., 2004; Hagiwara and Sekine, 2013; Kaji and Kitsuregawa, 2014; Morita et al., 2015; Fujinuma and Grissom II, 2017) or across multiple languages (Schuster and Nakajima, 2012; Sennrich et al., 2016; Kudo and Richardson, 2018).

Recently, this topic has been revisited in the context of Transformer-based models. Since the input strings are tokenized into tokens and then Transformers use token embeddings to map input strings to embeddings or real-valued vectors, which strings to consider as an token input to Transformers becomes important especially for multilingual models. The linguistic units considered in such Transformer-based models vary from words (Mikolov et al., 2013a), subwords (Peters et al., 2018; Devlin et al., 2019), characters (Peters et al., 2018; Clark et al., 2021), to bytes (Xue et al., 2021a; Tay et al., 2021b). There are trade-offs regarding which linguistic units to be considered as tokens. If words are used, then it will be easier to interpret and debug the outputs of the model. If I regard subwords, characters, or bytes as tokens, then (1) it will require less number of embedding parameters compared to words, and (2) it will be robust to out-of-vocabulary words (i.e., words which do not appear in the training data).

Inconsistency and Lack of Data across Languages When training NLP models, I use two types of data. One is the labeled, where the data is associated with a label (e.g., a class of a document for a document classification task), typically given by human annotators. However, these labeled data are often not available in low-resource languages. To overcome this issue, one common approach is **cross-lingual transfer**, i.e., training on a high-resource language and fine-tuning it on the target low-resource language (Klementiev et al., 2012; Ruder et al., 2017).

Another type of data used to train NLP models are unlabeled or raw texts. Pretrained language models can successfully exploit large unlabeled texts on languages. Therefore, pretraining is successful in languages such as English because it has a rich source of labeled texts created by linguists or crowd-source workers, and large amount of unlabeled texts. However, such a large amount of texts are not always available in all languages. Dataset availability is important because

pre-training contextualized embeddings and its quality is reported to decrease when having smaller data (e.g., 20K parallel sentences (Schuster et al., 2019)).

Scaling across Arbitrary Languages Pretrained cross-lingual language models are still under research on to what extent it can scale across languages. On cross-lingual dependency parsing, Ahmad et al. (2019b) report that when only using English labeled texts to fine-tune mBERT, consistently outperform static cross-lingual word embeddings in 28 target languages. However, for example Pires et al. (2019) report that cross-lingual learning across typologically distant languages, such as English and Japanese, is not successful compared to typologically closer languages (e.g., within the SVO languages). Eisenschlos et al. (2019) report that the accuracy of Multilingual BERT is inferior to static cross-lingual word embeddings by Ammar et al. (2016) for multiple languages on cross-lingual document classification. Finally, cross-lingual language models support languages seen during pretraining and do not guarantee to have high performance on languages not seen during pretraining (Chau et al., 2020). In Chapter 5, I analyze multilingual models with respect to different languages used during pretraining.

In the next chapter, I first focus on the issue of the inconsistency and lack of data across languages, especially on the lack of evaluation data, by proposing a resource-free metric to evaluate cross-lingual word embeddings.

Chapter 3

Evaluating Cross-Lingual Word Embeddings by Graph Modularity

One problem of cross-lingual learning using cross-lingual word embeddings is that when one of the target languages is a low-resource language, it often lacks the test data to evaluate trained models for the task of interest. Intuitively, when cross-lingual word embeddings are "mixed" in terms of languages, then cross-lingual transfer learning would likely be more successful. To quantify how well mixed a given cross-lingual word embedding is, I propose a graph-based intrinsic metric to evaluate the quality of cross-lingual word embeddings without requiring test data on the task of interest. The contents discussed in this chapter are largely based on Fujinuma et al. (2019).

3.1 Related Work on Evaluating Cross-Lingual Word Embeddings

Most work on evaluating cross-lingual embeddings focuses on extrinsic evaluation of downstream tasks (Upadhyay et al., 2016; Glavas et al., 2019). However, intrinsic evaluations are crucial since many low-resource languages lack annotations needed for downstream tasks. Thus, our goal is to develop an intrinsic measure that correlates with downstream tasks without using any external resources. This section summarizes existing work on intrinsic methods of evaluation for cross-lingual embeddings.

One widely used intrinsic measure for evaluating the coherence of monolingual embeddings is QVEC (Tsvetkov et al., 2015). Ammar et al. (2016) extend QVEC by using canonical correlation analysis (QVEC-CCA) to make the scores comparable across embeddings with different dimensions. However, while both QVEC and QVEC-CCA can be extended to cross-lingual word embeddings, they

are limited: they require external annotated corpora. This is problematic in cross-lingual settings since this requires annotation to be consistent across languages (Ammar et al., 2016).

Other internal metrics do not require external resources, but those consider only part of the embeddings. Conneau et al. (2018a) and Artetxe et al. (2018) use a validation metric that calculates similarities of cross-lingual neighbors to conduct model selection. Our approach differs in that I consider whether cross-lingual nearest neighbors are *relatively closer* than intra-lingual nearest neighbors.

Søgaard et al. (2018) use the similarities of intra-lingual nearest neighbors and compute graph similarity between two monolingual lexical subgraphs built by subsampled words in a bilingual lexicon. They further show that the resulting graph similarity has a high correlation with bilingual lexical induction on MUSE (Conneau et al., 2018a). However, their graph similarity still only uses intra-lingual similarities but not cross-lingual similarities.

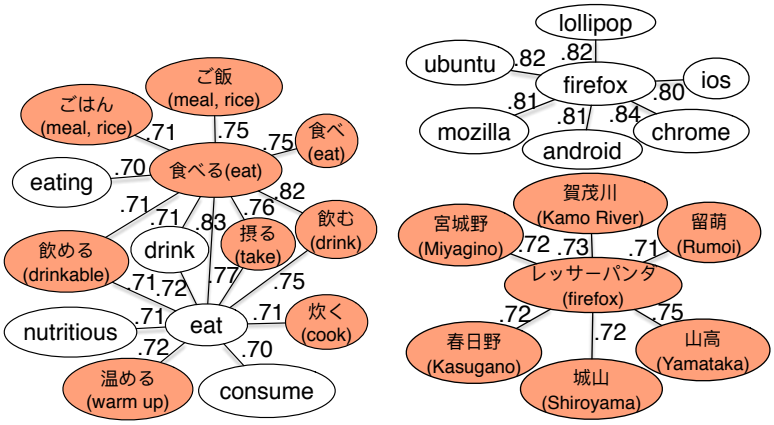
These existing metrics are limited by either requiring external resources or considering only part of the embedding structure (e.g., intra-lingual but not cross-lingual neighbors). In contrast, our work develops an intrinsic metric which is highly correlated with multiple downstream tasks but does not require external resources, and considers both intra- and cross-lingual neighbors.

3.2 Graph Modularity on Cross-Lingual Lexical Graphs

Graph modularity is originally defined to detect communities on a social network (Newman, 2003). In the context of evaluating cross-lingual word embeddings, I regard the language of each word (i.e., node) as the community of a node. Figure 3.1 shows an example of embedding-driven lexical graphs with different graph modualrities.

I now define graph modularity of an embedding-driven cross-lingual lexical graph. Formally, given a cross-lingual word embedding, the first step is to derive a lexical graph $G = (V, E)$ and its adjacency matrix A. The weights of an edge correspond to the cosine similarity $\cos(i, j)$ between words i and j in the cross-lingual embedding space.

Conceptually, the modularity of a lexical graph is the difference between the proportion of



(a) Low Modularity (b) High Modularity

Figure 3.1: An example of a cross-lingual word embedding-driven lexical graph with low modularity (languages mixed) and high modularity one using k-nearest neighbors of "eat" (left) and "firefox" (right) in English and Japanese.

edges in the graph that connect two nodes from the same language and the **expected** proportion of such edges in a randomly connected lexical graph. If edges were random, the number of edges starting from node i within the same language would be the degree of node i, $d_i = \sum_j A_{ij}$ for a weighted graph, following Newman (2004), times the proportion of words in that language. Summing over all nodes gives the expected number of edges within a language,

$$a_l = \frac{1}{2m} \sum_i d_i \mathbb{1}\left[g_i = l\right], \tag{3.1}$$

where m is the number of edges, g_i is the label of node i, and $\mathbb{1}\left[\cdot\right]$ is an indicator function that evaluates to 1 if the argument is true and 0 otherwise. Each node i has a label g_i indicating the word's language.

Next, I count the fraction of edges e_{ll} that connect words of the same language:

$$e_{ll} = \frac{1}{2m} \sum_{ij} A_{ij} \mathbb{1}\left[g_i = l\right] \mathbb{1}\left[g_j = l\right]. \tag{3.2}$$

Given L different languages, I calculate overall modularity Q by taking the difference between e_{ll} and a_l^2 for all languages:

$$Q = \sum_{l=1}^{L} (e_{ll} - a_l^2). \tag{3.3}$$

Since Q does not necessarily have a maximum value of 1, I normalize modularity:

$$Q_{norm} = \frac{Q}{Q_{max}}, \tag{3.4}$$

where $Q_{max} = 1 - \sum_{l=1}^{L}(a_l^2)$. The higher the modularity, the more words from the same language appear as nearest neighbors. Figure 3.1 shows the example of a lexical subgraph with low modularity (left, $Q_{norm} = 0.143$) and high modularity (right, $Q_{norm} = 0.672$).

Our hypothesis is that cross-lingual word embeddings with lower modularity will be more successful in downstream tasks. If this hypothesis holds, then modularity could be a useful metric for cross-lingual evaluation.

3.3 Cross-Lingual Mapping Methods Used in the Experiments

Three supervised (MSE, MSE+Orth, CCA) and the two unsupervised (MUSE, VECMAP) cross-lingual mappings explored in this chapter:[1]

Mean-squared error (MSE) MSE is using Eq. 2.2 introduced by Mikolov et al. (2013a), which minimizes the mean-squared error of bilingual entries in a training lexicon, to learn a projection between two embeddings. Here, I use the implementation by Artetxe et al. (2016).

MSE with orthogonal constraints (MSE+Orth) MSE+Orth is adding length normalization and orthogonal constraints to preserve the cosine similarities in the original monolingual embeddings (Xing et al., 2015). I further follow Artetxe et al. (2016) and further preprocess monolingual embeddings by mean centering.[2]

Canonical Correlation Analysis (CCA) CCA is the method proposed by Faruqui and Dyer (2014) which maps two monolingual embeddings into a shared space by maximizing the correlation between translation pairs in a bilingual lexicon.

MUSE by Conneau et al. (2018a) As introduced and explained in depth in Section 2.3.3.1, MUSE uses language-adversarial learning (Ganin et al., 2016) to induce the initial

[1] I use the implementations from original authors with default parameters unless otherwise noted.
[2] This preprocessing is equivalent to one round of iterative normalization (Zhang et al., 2019)

bilingual seed lexicon, followed by a refinement step, which iteratively solves the orthogonal Procrustes problem (Schönemann, 1966; Artetxe et al., 2017), aligning embeddings without an external bilingual lexicon. Like MSE+Orth, vectors are unit length and mean centered. Since MUSE is unstable (Artetxe et al., 2018; Søgaard et al., 2018), I report the best of five runs.

VECMAP by Artetxe et al. (2018) VECMAP proposed by Artetxe et al. (2018) induces an initial bilingual seed lexicon by aligning intra-lingual similarity matrices computed from each monolingual embedding. I report the best of five runs to address uncertainty from the initial dictionary.

3.4 Correlation to Cross-Lingual Downstream Tasks

I now show that graph modularity is a metric that can predict the downstream task performance without requiring test data in that task. Graph modularity on a cross-lingual lexical graph can capture an important characteristic of a cross-lingual word embedding, which is indicative of the downstream task performance.

3.4.1 Task 1: Document Classification

I now explore the correlation of modularity and accuracy in cross-lingual document classification. I classify documents from the Reuters RCV1 and RCV2 corpora (Lewis et al., 2004). Documents have one of four labels (Corporate/Industrial, Economics, Government/Social, Markets). I follow Klementiev et al. (2012), except I use all EN training documents and documents in each target language (DA, ES, IT, and JA) as tuning and test data. After removing out-of-vocabulary words, I split documents in target languages into 10% tuning data and 90% test data. Test data are 10,067 documents for DA, 25,566 for IT, 58,950 for JA, and 16,790 for ES. I exclude languages Reuters lacks: HU and AM. I use deep averaging networks (Iyyer et al., 2015, DAN) with three layers, 100 hidden states, and 15 epochs as our classifier. The DAN had better accuracy than averaged perceptron (Collins, 2002) in Klementiev et al. (2012).

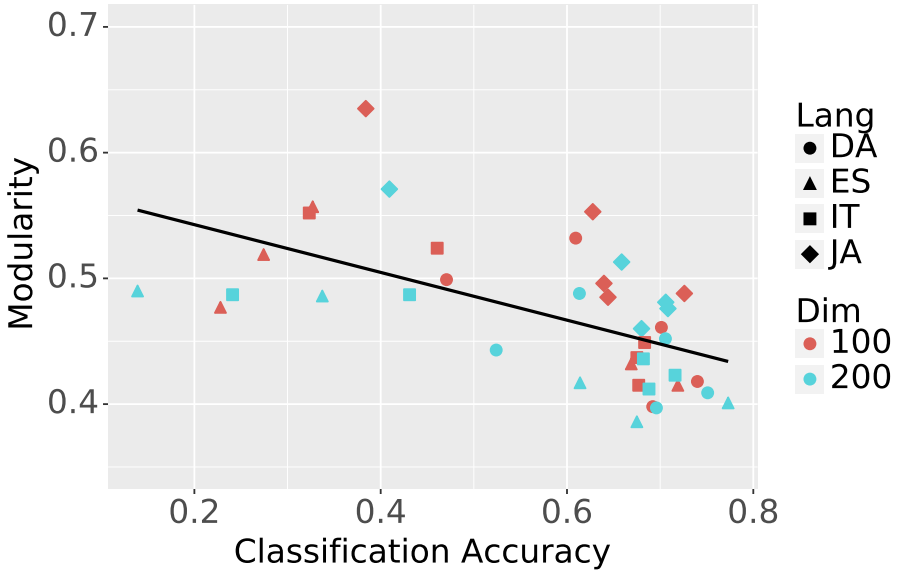


Figure 3.2: Document classification accuracy and modularity of cross-lingual word embeddings ($\rho = -0.665$): less modular cross-lingual mappings have higher accuracy.

	Method	Acc.	Modularity
Supervised	MSE	0.399	0.529
	CCA	0.502	0.513
	MSE+Orth	0.628	0.452
Unsupervised	MUSE	0.711	0.431
	VECMAP	0.643	0.432

Table 3.1: Average classification accuracy on (EN → DA, ES, IT, JA) along with the average modularity of five cross-lingual word embeddings. MUSE has the best accuracy, captured by its low modularity.

Results I report the correlation value computed from the data points in Figure 3.2. Spearman's correlation between modularity and classification accuracy on all languages is $\rho = -0.665$. Within each language pair, modularity has a strong correlation within EN-ES embeddings ($\rho = -0.806$), EN-JA ($\rho = -0.794$), EN-IT ($\rho = -0.784$), and a moderate correlation within EN-DA embeddings ($\rho = -0.515$). MUSE has the best classification accuracy (Table 3.1), reflected by its low modularity.

Error Analysis A common error in EN → JA classification is predicting Corporate/Industrial for documents labeled Markets. One cause is documents with 終値 "closing price"; this has few market-based English neighbors (Table 3.2). As a result, the model fails to transfer across lan-

市場 “market”	終値 “closing price”
新興 “new coming”	上げ幅 “gains”
market	株価 “stock price”
markets	年初来 “yearly”
軟調 “bearish”	続落 “continued fall”
マーケット “market”	月限 “contract month”
活況 “activity”	安値 “low price”
相場 “market price”	続伸 “continuous rise”
底入 “bottoming”	前日 “previous day”
為替 “exchange”	先物 “futures”
ctoc	小幅 “narrow range”

Table 3.2: Nearest neighbors in an EN-JA embedding. Unlike the JA word “market”, the JA word “closing price” has no EN vector nearby.

guages.

3.4.2 Task 2: Bilingual Lexical Induction (BLI)

Our second downstream task explores the correlation between modularity and bilingual lexical induction (BLI). I evaluate on the test set from Conneau et al. (2018a), but I remove pairs in the seed lexicon from Rolston and Kirchhoff (2016). The result is 2,099 translation pairs for ES, 1,358 for IT, 450 for DA, and 973 for JA. I report precision@1 (P@1) for retrieving cross-lingual nearest neighbors by cross-domain similarity local scaling (Conneau et al., 2018a, CSLS).

Results Although this task ignores intra-lingual nearest neighbors when retrieving translations, modularity still has a high correlation ($\rho = -0.785$) with P@1 (Figure 3.3). MUSE and VECMAP beat the three supervised methods, which have the lowest modularity (Table 3.3). P@1 is low compared to other work on the MUSE test set (e.g., Conneau et al. (2018a)) because I filter out translation pairs which appeared in the large training lexicon compiled by Rolston and Kirchhoff (2016), and the raw corpora used to train monolingual embeddings are relatively small compared to Wikipedia.

3.4.3 Task 3: Document Retrieval in Low-Resource Languages

As a third downstream task, I turn to an important task for low-resource languages: lexicon expansion (Gupta and Manning, 2015; Hamilton et al., 2016) for document retrieval. Specifically,

	Method	P@1	Modularity
Supervised	MSE	7.30	0.529
	CCA	3.06	0.513
	MSE+Orth	10.57	0.452
Unsupervised	MUSE	11.83	0.431
	VECMAP	12.92	0.432

Table 3.3: Average precision@1 on (EN → DA, ES, IT, JA) along with the average modularity of the cross-lingual word embeddings trained with different methods. VECMAP scores the best P@1, which is captured by its low modularity.

I start with a set of EN seed words relevant to a particular concept, then find related words in a target language for which a comprehensive bilingual lexicon does not exist. We focus on the disaster domain, where events may require immediate NLP analysis (e.g., sorting SMS messages to first responders).

I induce keywords in a target language by taking the n nearest neighbors of the English seed words in a cross-lingual word embedding. I manually select sixteen disaster-related English seed words from Wikipedia articles, "*Natural hazard*" and "*Anthropogenic hazard*". Examples of seed terms include "earthquake" and "flood". Using the extracted terms, I retrieve disaster-related documents by keyword matching and assess the coverage and relevance of terms by area under the precision-recall curve (AUC) with varying n.

Test Corpora As positively labeled documents, I use documents from the LORELEI project (Strassel and Tracey, 2016) containing any disaster-related annotation. There are 64 disaster-related documents in Amharic, and 117 in Hungarian. I construct a set of negatively labeled documents from the Bible; because the LORELEI corpus does not include negative documents and the Bible is available in all our languages (Christodouloupoulos and Steedman, 2015), I take the chapters of the gospels (89 documents), which do not discuss disasters, and treat these as non-disaster-related documents.

Results Modularity has a moderate correlation with AUC ($\rho = -0.378$, Table 3.4). While modularity focuses on the entire vocabulary of cross-lingual word embeddings, this task focuses on a small, specific subset—disaster-relevant words—which may explain the low correlation compared to BLI or document classification.

Lang.	Method	AUC	Mod.
AM	MSE	0.578	0.628
	CCA	0.345	0.501
	MSE+Orth	0.606	0.480
	MUSE	0.555	0.475
	VECMAP	0.592	0.506
HU	MSE	0.561	0.598
	CCA	0.675	0.506
	MSE+Orth	0.612	0.447
	MUSE	0.664	0.445
	VECMAP	0.612	0.432
Spearman Correlation ρ		-0.378	

Table 3.4: Correlation between modularity and area-under-the-curve (AUC) on document retrieval. It shows a moderate correlation to this task.

3.5 Use Case: Model Selection for MUSE

A common use case of intrinsic measures is model selection. I focus on MUSE (Conneau et al., 2018a) since it is unstable, especially on distant language pairs (Artetxe et al., 2018; Søgaard et al., 2018; Hoshen and Wolf, 2018) and therefore requires an effective metric for model selection. MUSE uses a validation metric in its two steps: (1) the language-adversarial step, and (2) the refinement step. First the algorithm selects an optimal mapping W using a validation metric, obtained from language-adversarial learning (Ganin et al., 2016). Then the selected mapping W from the language-adversarial step is passed on to the refinement step (Artetxe et al., 2017) to re-select the optimal mapping W using the same validation metric after each epoch of solving the orthogonal Procrustes problem (Schönemann, 1966).

Normally, MUSE uses an intrinsic metric, CSLS of the top 10K frequent words (Conneau et al., 2018a, CSLS-10K). What if modularity is used instead? To test modularity as a validation metric for MUSE, we compute modularity on the lexical graph of 10K most frequent words (Mod-10K; I use 10K for consistency with CSLS on the same words) after each epoch of the adversarial step and the refinement step and select the best mapping.

The important difference between these two metrics is that Mod-10K considers the relative similarities between intra- and cross-lingual neighbors, while CSLS-10K only considers the similari-

Family	Lang.	csls-10K		Mod-10K	
		Avg.	Best	Avg.	Best
Germanic	DA	**52.62**	**60.27**	52.18	60.13
	DE	**75.27**	**75.60**	75.16	75.53
Romance	ES	**74.35**	83.00	74.32	83.00
	IT	78.41	78.80	**78.43**	78.80
Indo-Iranian	FA	**27.79**	33.40	27.77	33.40
	HI	25.71	33.73	**26.39**	**34.20**
	BN	0.00	0.00	**0.09**	**0.87**
Others	FI	4.71	47.07	4.71	47.07
	HU	**52.55**	54.27	52.35	**54.73**
	JA	18.13	49.69	**36.13**	49.69
	ZH	5.01	37.20	**10.75**	37.20
	KO	16.98	20.68	**17.34**	**22.53**
	AR	15.43	33.33	**15.71**	**33.67**
	ID	67.69	68.40	**67.82**	68.40
	VI	0.01	0.07	0.01	0.07

Table 3.5: BLI results (precision@1 ×100%) from EN to each target language with different validation metrics for MUSE: default (CSLS-10K) and modularity (Mod-10K). I report the average (Avg.) and the best (Best) from ten runs with ten random seeds for each validation metric. **Bold** values are mappings that are not shared between the two validation metrics. Mod-10K improves the robustness of MUSE on distant language pairs.

ties of cross-lingual nearest neighbors.[3]

Experiment Setup The pre-trained fastText vectors (Bojanowski et al., 2017) to be comparable with the prior work. Following Artetxe et al. (2018), all vectors are unit length normalized, mean centered, and then unit length normalized. The test lexicon is from Conneau et al. (2018a). All experiments run ten times with the same random seeds and hyperparameters but with different validation metrics. Since MUSE is unstable on distant language pairs (Artetxe et al., 2018; Søgaard et al., 2018; Hoshen and Wolf, 2018), I test it on English to languages from diverse language families: Indo-European languages such as Danish (DA), German (DE), Spanish (ES), Farsi (FA), Italian (IT), Hindi (HI), Bengali (BN), and non-Indo-European languages such as Finnish (FI), Hungarian (HU), Japanese (JA), Chinese (ZH), Korean (KO), Arabic (AR), Indonesian (ID), and Vietnamese (VI).

Results Table 3.5 shows precision@1 on BLI for each target language using English as

[3] Another difference is that k-nearest neighbors for CSLS-10K is $k = 10$, whereas Mod-10K uses $k = 3$. However, using $k = 3$ for CSLS-10K leads to worse results; I therefore only report the result on the default metric i.e., $k = 10$.

the source language. Mod-10K improves precision@1 over the default validation metric in diverse languages, especially on the average precision@1 for non-Germanic languages such as JA (+18.00%) and ZH (+5.74%), and the best precision@1 for KO (+1.85%). These language pairs include pairs (EN-JA and EN-HI), which are difficult for MUSE (Hoshen and Wolf, 2018). Improvements in JA come from selecting a better mapping during the refinement step, which the default validation misses. For ZH, HI, and KO, the improvement comes from selecting better mappings during the adversarial step. However, modularity does not improve on all languages (e.g., VI) that are reported to fail by Hoshen and Wolf (2018).

3.6 Comparison to Other Intrinsic Evaluation Metrics

The experiments so far show that modularity captures whether an embedding is useful, which suggests that modularity could be used as an intrinsic evaluation or validation metric. Here, I investigate whether modularity can capture **distinct** information compared to existing evaluation measures: QVEC-CCA (Ammar et al., 2016), CSLS (Conneau et al., 2018a), and cosine similarity between translation pairs.

When fitting a linear regression model to predict the classification accuracy, four intrinsic measures are used: QVEC-CCA, CSLS, average cosine similarity of translations, and modularity. I ablate each of the four measures, fitting linear regression with standardized feature values, for two target languages (IT and DA) on the task of cross-lingual document classification. I limit to IT and DA because aligned supersense annotations to EN ones (Miller et al., 1993), required for QVEC-CCA are only available in those languages (Montemagni et al., 2003; Martínez Alonso et al., 2015; Martınez Alonso et al., 2016; Ammar et al., 2016). I standardize the values of the four features before training the regression model. Omitting modularity hurts accuracy prediction on cross-lingual document classification substantially, while omitting the other three measures has smaller effects (Figure 3.4). Thus, modularity complements the other measures and is more predictive of classification accuracy.

3.7 Discussion: What Modularity Can and Cannot Do

In this chapter, I focus on modularity as a diagnostic tool: it is cheap and effective at discovering which embeddings are likely to falter on downstream tasks. Thus, practitioners should consider including it as a metric for evaluating the quality of their embeddings. Additionally, I believe that modularity could serve as a useful prior for the algorithms that learn cross-lingual word embeddings: during learning prefer updates that avoid increasing modularity if all else is equal.

Nevertheless, there are limitations of modularity. Consider the following cross-lingual word embedding "algorithm": for each word, select a random point on the unit hypersphere. This is a horrible distributed representation: the position of words' embedding has no relationship to the underlying meaning. Nevertheless, this representation will have very low modularity. Thus, while modularity can identify bad embeddings, once vectors are well mixed, this metric—unlike QVEC or QVEC-CCA—cannot identify whether the meanings make sense.

This chapter proposes the evaluation method of cross-lingual using graph modularity to evaluate cross-lingual word embeddings. However, so far the use of graph modualarity is limited to be included as a validation metric in the training pipeline and not included as part of the objective function. The next chapter introduces the work on improving cross-lingual word embeddings by modifying objective functions and using a task-specific model for cross-lingual transfer learning. I first introduce the approach to directly optimize the graph modularity metric inspired by the results from this chapter.

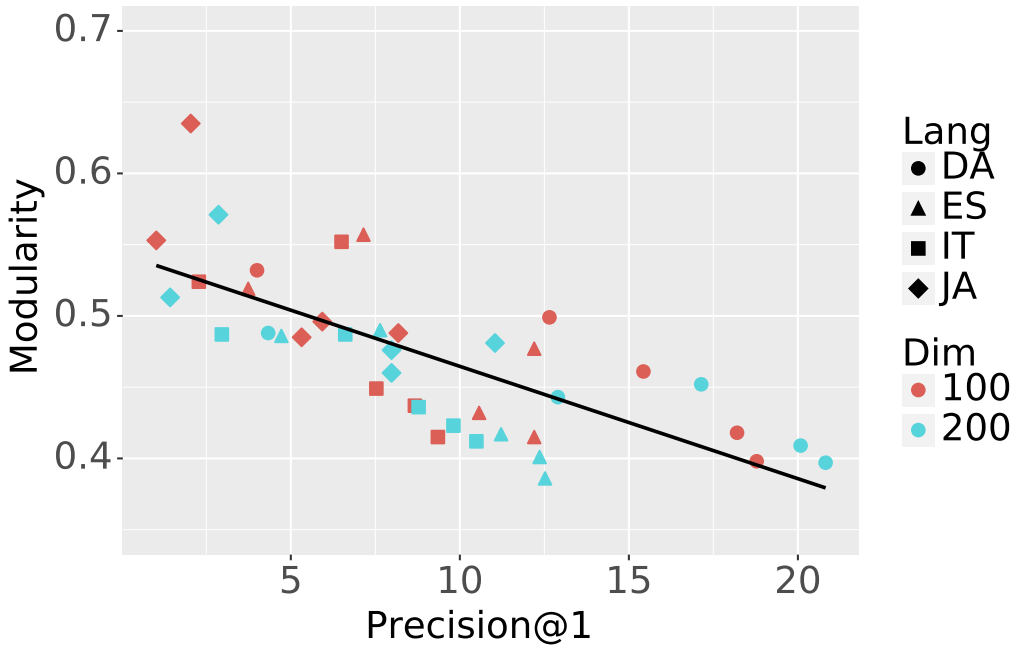


Figure 3.3: Bilingual lexical induction results and modularity of cross-lingual word embeddings ($\rho = -0.789$): lower modularity means higher precision@1.

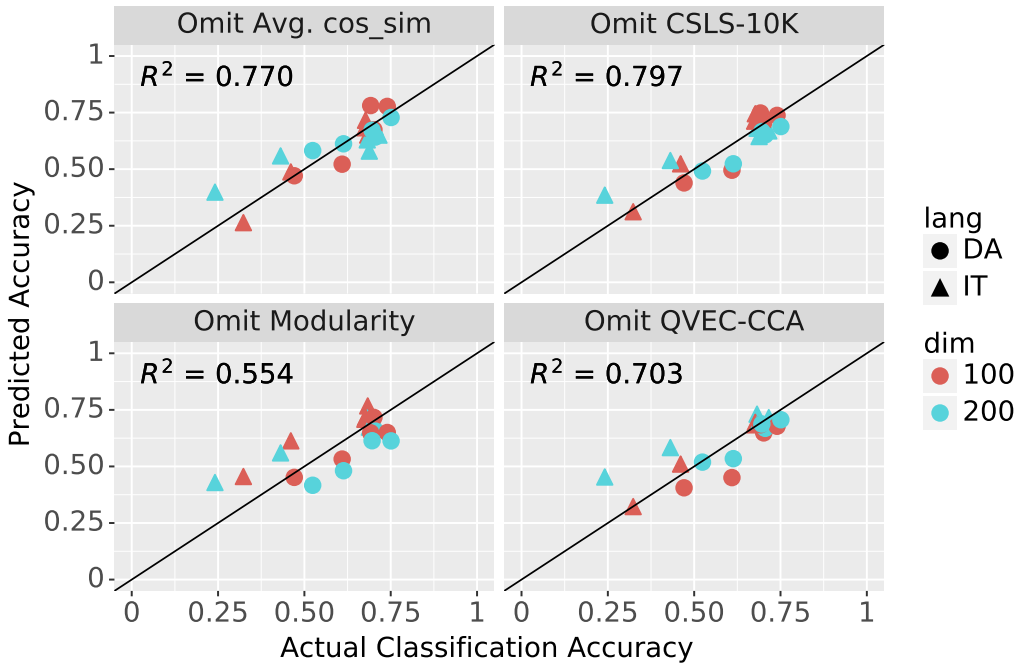


Figure 3.4: I predict the cross-lingual document classification results for DA and IT from Figure 3.2 using three out of four evaluation metrics. Ablating modularity causes by far the largest decrease ($R^2 = 0.814$ when using all four features) in R^2, showing that it captures information complementary to the other metrics.

Chapter 4

Improving Cross-Lingual Learning

This chapter turns from evaluating cross-lingual representations to improving cross-lingual representations. The contents in this chapter are largely based on Zhang et al. (2020) and Fujinuma and Hagiwara (2021).[1]

Section 4.1 introduces graph auto-encoder (GAE) to incorporate graph modularity in the loss function while training cross-lingual word embeddings with mixed results. Section 4.2 further prunes the architecture of GAE and focus on the graph encoder, which shows consistent improvement in the bilingual lexical induction task. Section 4.3 experiments cross-lingual dependency parsing with a simple approach to improve cross-lingual word embeddings using a bilingual lexicon. Finally, Section 4.4 discusses a way to improve in a task-specific setting using readability estimation for second language learners as an example.

4.1 Optimizing Graph Modularity Using Graph Auto-Encoders

This section discusses the results on incorporating graph modularity as a loss function when training cross-lingual word embeddings.

4.1.1 Graph Auto-Encoders (GAE)

Inspired from Faruqui et al. (2015)'s work on using graph-based approach to refine word embeddings, this section introduces a method to refine a given cross-lingual word embedding using

[1] Section 4.3 is based on Zhang et al. (2020) and Section 4.4 is based on Fujinuma and Hagiwara (2021).

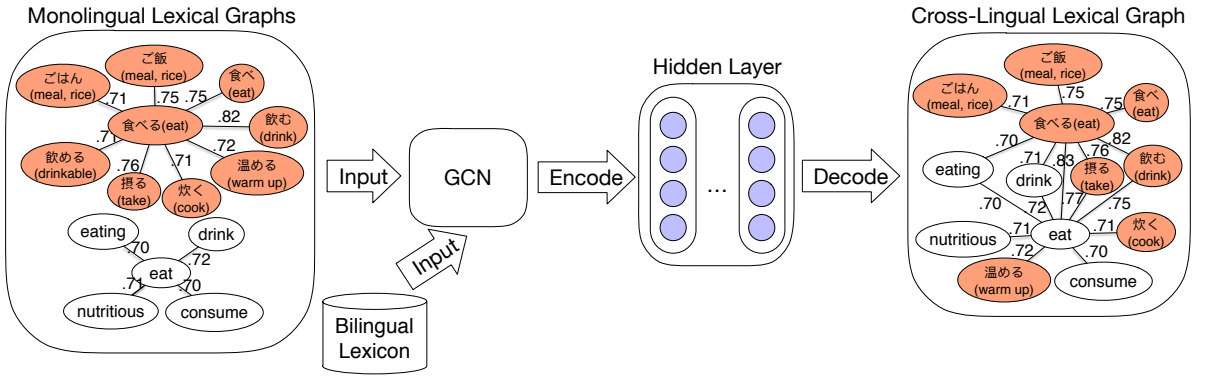


Figure 4.1: Overview of the pipeline to train and refine cross-lingual word embeddings using graph auto-encoder (GAE). I focus and explore the importance of using refinement methods to pre-trained cross-lingual word embeddings.

a graph auto-encoder (**GAE**). One main difference between existing retro-fitting methods is using the local neighborhood similarities of pre-trained embeddings as the only supervision signals. I introduce a preprocessing step that converts word embeddings into a lexical graph, and then explain each component of our proposed method: Supervision by a bilingual lexicon, and graph auto-encoder.

Supervision by Bilingual Lexicon I follow Mikolov et al. (2013a) and include the supervision from a bilingual lexicon by mean squared loss of given aligned translation pairs. I further include oracle aligned node pairs $(s, t) \in T_{train}$ to the overall loss function:

$$L_{align} = \frac{1}{n} \sum_{(s,t) \in T_{train}}^{n} \|\boldsymbol{h}_s - \boldsymbol{h}_t\|_2^2 \tag{4.1}$$

where L_{align} is the mean-squared loss for aligned vectors given by a bilingual lexicon, $\boldsymbol{h}_s$ and $\boldsymbol{h}_t$ are the corresponding hidden layer for a source word s and a target word t.

Preserving Local Neighborhood of Embedding-Driven Lexical Graphs

Given a cross-lingual word embedding C outputted from exisiting methods, we first convert it into a k-nearest neighbor (kNN) lexical graph $G = (V, E)$. Nodes V are the vocabularies in a cross-lingual word embedding, edges E are weighted by the sigmoid of the cosine similarity between words i.e., $e_{ij} = \sigma(\cos(v_i, v_j))$. This pre-processing step avoids the non-differentiablity issue of kNN Plötz and Roth (2018) during optimization.

Given a cross-lingual lexical graph, graph auto-encoders (GAE) embed every node in a graph into a real-valued vector such that it can reconstruct the original input graph. I use the method by Kipf and Welling (2016). Given an adjacency matrix A of the cross-lingual lexical graph G, graph convolutional network (Kipf and Welling, 2017, GCN) with parameters $W \in \mathbb{R}^{|V| \times h}$ is

$$\boldsymbol{h} = \sigma_e(WXA_{norm}) \tag{4.2}$$

where $\boldsymbol{h} \in \mathbb{R}^{|V| \times h}$ is the embeddings, σ_e is the activation function for the encoder, W is a linear transformation matrix, X is the feature representation for each node, and $A_{norm} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ is the normalized adjacency matrix of A, where D is the degree matrix of A. In practice, we use $X = I$ and identity function for the activation function σ_e.

The reconstructed adjacency matrix $\hat{A}$ is decoded from the embeddings $\boldsymbol{h} \in \mathbb{R}^{|V| \times h}$ from Eq. 4.2, i.e.,

$$\hat{A} = \sigma_d(\boldsymbol{h}\boldsymbol{h}^T) \tag{4.3}$$

where σ_d is the activation function for the decoder applied to each element. In practice, we use the sigmoid function to have an output of $[0, 1]$. $\boldsymbol{h}\boldsymbol{h}^T$ enforces $\hat{A}$ to become a symmetric matrix, which preserves the property of the original input adjacency matrix A.

The reconstruction loss function $L_{reconst}$ is defined as the cross-entropy loss between the original input matrix A and the reconstructed adjacency matrix $\hat{A}$, i.e,

$$\begin{aligned} L_{reconst} = -\frac{1}{|E|} \sum_{ij}^{|E|} & A_{ij} \log\Big(\hat{A}_{ij}\Big) \\ & + (1 - A_{ij}) \log\Big(1 - \hat{A}_{ij}\Big) \end{aligned} \tag{4.4}$$

which predicts the weight (i.e., the similarity of word i and j) of each edge $\hat{A}_{ij}$ in the reconstructed adjacency matrix $\hat{A}$. The reconstruction loss also exploits information of the edges which do not exist (i.e., $A_{ij} = 0$), and enforces those nodes to have weights close to zero in the reconstructed adjacency matrix A_{ij}.

4.1.1.1 Encouraging Cross-Lingual Mixture by Modularity Regularizer

The reconstruction loss in the previous section only preserves the intra-lingual similarity, and there is a risk of producing cross-lingual word embeddings which overly clusters intra-lingual neighbors. To encourage the two monolingual embeddings to be mixed, we further add a regularizer based on modularity of a cross-lingual lexical graph. Modularity Q (Newman, 2006b) is a graph-based metric which quantifies the strength of the clusters of given communities (i.e., languages in our case) of a given graph. Given number of languages l, modularity is computed as

$$Q = \mathrm{Tr}\big(H^T B H\big) \tag{4.5}$$

where $H \in \{0,1\}^{|V| \times l}$ is the community indicator matrix, which is a one-hot representation of the language of each node, and B is the modularity matrix (Newman, 2006b) where each element $B_{ij} \in \mathbb{R}$ of B is

$$B_{ij} = \hat{A}_{ij} - \frac{k_i k_j}{2m}, \tag{4.6}$$

where k_i is the degree of node i (which counts both intra- and cross-lingual edges), and m is the total number of edges in $\hat{A}$. Intuitively, this value shows the difference between the existence of an intra-lingual edge $\hat{A}_{ij}$ and the expected number of intra-lingual edges $\frac{k_i k_j}{2m}$ based only on the degree of each node. The number of languages l is two for training a bilingual word embedding.

When a reconstructed lexical graph has negative modularity, the graph becomes bipartite, i.e., no intra-lingual edges and only cross-lingual edges exists (Newman, 2006a). However, we want the cross-lingual neighbors to appear as much as intra-lingual neighbors do, and we do not want nearest neighbor graphs derived from a cross-lingual embedding to be bipartite. To penalize a cross-lingual lexical graph being a bipartite graph, we add the L2 norm on top of the modularity regularizer

$$L_{mod} = \|Q\|_2. \tag{4.7}$$

Finally, we define the overall loss L as the sum of reconstruction loss $L_{reconst}$, supervision loss L_{align} with regularization parameter λ with regularization parameter λ, and modularity regularizer

Method	EN-ES	EN-DE	EN-TR	EN-KO	EN-JA	EN-FI	EN-AM	EN-TI
Procrustes	**37.9**	25.0	9.8	5.3	12.5	9.5	2.1	*2.6
RCSLS	37.7	**25.9**	**13.5**	11.6	*15.5	**11.9**	*3.0	**4.0**
GAE	22.5	17.3	11.6	16.5	20.0	8.7	**3.4**	**4.0**
GAE+mod	25.8	20.2	**13.5**	**17.2**	**20.9**	10.4	3.1	**4.0**

Table 4.1: BLI results (**Precision@1**) on the MUSE test using fastText trained on the corpora from the Leipzig corpora collection (Goldhahn et al., 2012) and the LORELEI dataset (Strassel and Tracey, 2016). The values with (*) are obtained by nearest neighbor retrieval using cosine similarity, which gave higher score than using CSLS.

L_{mod} with regularization parameter λ:

$$L = L_{reconst} + L_{align} + \lambda L_{mod}. \tag{4.8}$$

4.1.2 Experiments

4.1.2.1 Results on Monolingual Embeddings Trained on Small Corpora

Table 4.1 shows the BLI results using all resources available i.e., full size corpus and full size training lexicon. There is a trend that GAE is robust on etymologically distant language pairs (English to Turkish, Korean, Japanese, Finnish, Amharic, and Tigrinya). For close language pairs (English-Spanish and English-German), the projection-based approaches outperform GAE.

Comparing with and without adding modularity regularizer to GAE improves on all language pairs except for English-Amharic (Table 4.1). This is consistent with the results on the synthetic data.

4.1.2.2 Results on Monolingual Embeddings Trained on Wikipedia

To further confirm that the results on low-resource languages hold on monolingual embeddings trained on other corpora, we further test our methods using the pre-trained fastText vectors trained on Wikipedia provided by Bojanowski et al. (2017).

The trend is also similar using the Wikipedia vectors. In Table 4.2, projection-based methods outperforms on English-Spanish, but GAE and GAE+mod outperform on English to low-resource

Method	EN-ES	EN-AM	EN-TI
Procrustes	81.1	4.9	*2.8
RCSLS	**81.8**	*4.0	4.9
GAE	21.7	8.1	**10.5**
GAE+mod	12.1	**8.8**	8.3

Table 4.2: BLI results (Precision@1) on the MUSE test set pre-trained fastText vectors trained on Wikipedia provided by Bojanowski et al. (2017).

languages (Amharic[2] and Tigrinya).

In this section, we see mixed results using GAE in Table 4.1 and Table 4.2 that do not show consistant improvement over commonly-used cross-lingual word embedding methods such as Orthogonal Procrustes and RCSLS. To further dive deep into the results, we prune GAE and focus on its graph encoder part in the next section.

4.2 Improving Cross-Lingual Word Embeddings by Linear Graph Encoder

Existing pre-trained methods (language model (Devlin et al., 2019) or skip-gram models (Mikolov et al., 2013b)) encode useful structure (e.g., words with similar meanings are embededded close to each other) in the pre-trained parameters, both intra- and cross-lingually, in the pre-training step. One problem of fine-tuning these parameters for downstream tasks is that it can easily break its useful structure.

One key solution is to preserve the local neighborhood structure. Nakashole and Flauger (2018) report that a single linear mapping to map from language embedding to another embedding is not enough to fully exploit the training bilingual dictionary while learning cross-lingual word embeddings. Their work further supports the focus on the local structure of embeddings rather than the global one when learning static cross-lingual word embeddings. However, Nakashole (2018) also reports that training a single linear mapping from one embedding space to another still achieves better results on the benchmark dataset.

To preserve the local neighborhood structure in cross-lingual embeddings, we propose a

[2] Amharic Wikipedia is around 400 times smaller than the English one `https://meta.wikimedia.org/wiki/List_of_Wikipedias`.

method which adds **local structural bias** during training to refine static cross-lingual word embeddings.

4.2.1 Graph Convolutional Network (GCN) and Linear Graph Encoder

Kipf and Welling (2017) proposed an approach to apply a neural network on graphs. GCN has been used in several domains, including texts (Vashishth et al., 2019). Given a set of vertices V, a set of edges E, and a graph $G = (V, E)$ and its adjacency matrix $A \in \mathbb{R}^{|V|\times|V|}$, each layer $\boldsymbol{h}_l \in \mathbb{R}^{|V|\times h_l}$ in GCN is computed using the previous layer $\boldsymbol{h}_{l-1}$ as

$$\boldsymbol{h}_l = \sigma(A\boldsymbol{h}_{l-1}W_l) \tag{4.9}$$

where σ is a non-linear activation function, and $W_l \in \mathbb{R}^{h_{l-1}\times h_l}$ is the parameter matrix for the lth layer. The first layer $\boldsymbol{h}_1$ is computed as

$$\boldsymbol{h}_1 = \sigma(AXW_1) \tag{4.10}$$

$W_1 \in \mathbb{R}^{x\times h_1}$ is the parameter matrix, $X \in \mathbb{R}^{|V|\times x}$ is the feature representation matrix with x dimensions for each node in V.

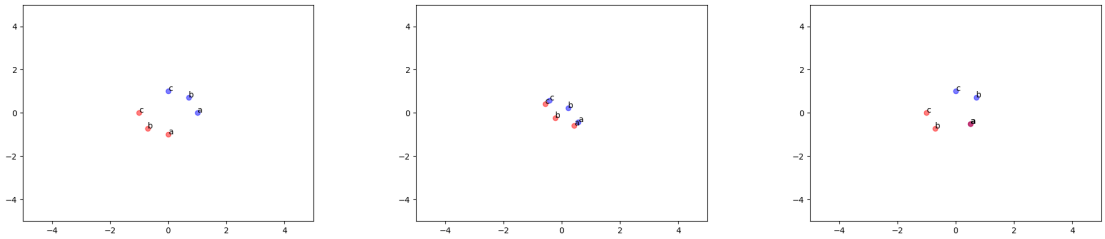
Because Salha et al. (2019) report that non-linear activation is not always necessary, I use the simplified version of GCN called linear graph encoder (Salha et al., 2019) i.e., activation function being an identity function

$$\boldsymbol{h}_l = AW, \tag{4.11}$$

to encode structural bias during training.

4.2.2 Encoding Local Structural Bias

I now introduce how to encode local structural bias into the model. Salha et al. (2019) discuss that the linear graph encoder (Eq. 4.11) , i.e. GCN without a non-linear activation function, is sufficient to embed nodes and conduct link prediction and node clustering. In the preliminary experiments on comparing GCN and the linear graph encoder, GCN slightly performed worse



(a) Synthetic pre-trained cross-lingual word embeddings.

(b) Retrofitted with the linear graph encoder.

(c) Retrofitted without the linear graph encoder.

Figure 4.2: Example results on with and without the linear graph encoder on the synthetic data. Retrofitting minimizes the Euclidean distance between the vector $\mathbf{v}'$ assigned to the red point a and the vector $\mathbf{v}'$ assigned to the blue point a.

when retrofitting cross-lingual word embeddings, and therefore, we use the linear graph encoder for the rest of this chapter.

I encode structural bias by using a linear graph encoder using a k-nearest neighbor graph of the pre-trained embeddings. Let $\mathbf{X}$ be the pre-trained embedding matrix and $\mathbf{X}'$ be the refined embedding embedded by modifying the original embedding matrix $\mathbf{X}$ by the perturbation vectors AW i.e.,

$$\mathbf{X}' = \mathbf{X} + AW \tag{4.12}$$

where A is the adjacency matrix and W are the parameters. I freeze $\mathbf{X}$ and A during the training, and only optimize W.

For retrofitting the static cross-lingual word embeddings, we use the loss function

$$L_{\text{retrofit}}(W) = \sum_{(i,j)\in D} \left\|\mathbf{X}'_i - \mathbf{X}'_j\right\|_2 \tag{4.13}$$

where i and j are the translation pair words in the training bilingual dictionary.

4.2.2.1 Experiment on Synthetic Data

I first experiment the effectiveness of the linear graph encoder on refining static cross-Lingual word embeddings using a training bilingual dictionary. Figure 4.2 shows the result of using the

Methods	FR	DE	JA	ZH	KO	AR	TH
OrthProc	84.23	82.01	63.32	59.08	59.64	55.84	46.39
+Retrofit	80.49	79.33	61.53	63.58	60.73	47.79	46.13
+Retrofit+graph enc.	**85.04**	**82.49**	**65.30**	**64.70**	**63.09**	**57.75**	**48.18**
RCSLS	84.84	83.52	68.04	71.07	66.18	59.39	45.95
+Retrofit	78.64	78.11	58.18	67.51	60.82	44.16	41.67
+Retrofit+graph enc.	**85.18**	**84.75**	**69.41**	**72.85**	**67.27**	**60.17**	**47.46**

Table 4.3: BLI results (Precision@1) on each target language with and without using linear graph encoder (graph enc.). Without the linear graph encoder, it shows consistent decrease in the BLI results after retrofitting the pre-trained cross-lingual word embeddings.

linear graph encoder on the synthetic data. This synthetic experimental result shows that it can retrofit using the training dictionary while maintaining the local structure.

4.2.2.2 Experiment on Real Data

I now experiment on the real data. For the pre-trained monolingual word embeddings, we use the one by Grave et al. (2018). For the training and test bilingual dictionaries, we use the MUSE lexicon (Conneau et al., 2018a). Using the pre-trained monolingual word embeddings and training bilingual dictionaries, I train cross-lingual word embeddings using two different methods: Orthogonal Procrustes (Xing et al., 2015) and Relaxed CSLS (Joulin et al., 2018, RCSLS).

Table 4.3 shows the BLI results on retrofitted cross-lingual word embeddings with and without using the linear graph encoder. I use the maximum size of training bilingual dictionaries available in the MUSE dataset (Conneau et al., 2018a). Precision@1 consistently decreases on the BLI results without the linear graph encoder, and therefore, it shows that local structural bias is a good bias.

4.3 Simple Approach: Overfitting to a Bilingual Lexicon

In the previous section, the simple retrofitting decreases the accuracy on the bilingual lexical induction task. However, Glavas et al. (2019) report that the results on the bilingual lexical induction task do not necessarily correlate with the results on downstream tasks. In this section, we further examine the effect of a simple retrofitting method on the cross-lingual dependency

parsing task.

4.3.1 Overfitting and its Effect on Dependency Parsing

Here, a simple approach is explored to further exploit a bilingual lexicon: first train projection-based CLWE and then retrofit to the training dictionary (Figure 4.1). Retrofitting was originally introduced for refining monolingual word embeddings with synonym constraints from a lexical ontology (Faruqui and Dyer, 2014). For CLWE, I retrofit using the training dictionary $\mathcal{D}$ as the ontology.

Intuitively, retrofitting pulls translation pairs closer while minimizing deviation from the original CLWE. Let X' and Z' be CLWE trained by a projection-based method, where $X' = WX$ are the projected source embeddings and $Z' = Z$ are the target embeddings. The new CLWE $\hat{X}$ and $\hat{Z}$ are learned by minimizing

$$L = L_a + L_b, \tag{4.14}$$

where L_a is the squared distance between the updated CLWE from the original CLWE:

$$L_a = \alpha\|\hat{X} - X'\|^2 + \alpha\|\hat{Z} - Z'\|^2, \tag{4.15}$$

and L_b is the total squared distance between translations in the dictionary:

$$L_b = \sum_{(i,j)\in\mathcal{D}} \beta_{ij}\|\hat{x}_i - \hat{z}_j\|^2. \tag{4.16}$$

The same α and β as Faruqui and Dyer (2014) are used to balance the two objectives.

Retrofitting tends to overfit. If α is zero, minimizing L_b collapses each training pair to the same vector. Thus, all training pairs are perfectly aligned. In practice, I use a non-zero α for regularization. If the training dictionary covers predictive words, I expect retrofitting to improve downstream task accuracy.

4.3.2 Retrofitting to Synthetic Dictionary

While retrofitting brings pairs in the training dictionary closer, the updates may also separate translation pairs outside of the dictionary because retrofitting ignores words outside the training dictionary. This can hurt both BLI test accuracy and downstream task accuracy. In contrast,

projection-based methods underfit but can discover translation pairs outside the training dictionary. To keep the original CLWE's correct translations, we retrofit to both the training dictionary and a synthetic dictionary induced from CLWE (orange, Figure 4.1).

Early work induces dictionaries from CLWE through nearest-neighbor search (Mikolov et al., 2013a). I instead use cross-domain similarity local scaling (Conneau et al., 2018a, CSLS), a translation heuristic more robust to hubs (Dinu et al., 2014) (a word is the nearest neighbor of many words). I build a synthetic dictionary $\mathcal{D}'$ with word pairs that are mutual CSLS nearest neighbors. I then retrofit the CLWE to a combined dictionary $\mathcal{D} \cup \mathcal{D}'$. The synthetic dictionary keeps closely aligned word pairs in the original CLWE, which sometimes improves downstream models.

4.3.3 Experiments

I retrofit three projection-based CLWE to their training dictionaries and synthetic dictionaries and evaluate on the dependency parsing task.

4.3.3.1 Embeddings and Dictionaries

I align English embeddings with six target languages: German (DE), Spanish (ES), French (FR), Italian (IT), Japanese (JA), and Chinese (ZH). I use 300-dimensional fastText vectors trained on Wikipedia and Common Crawl (Grave et al., 2018). I lowercase all words, only keep the 200K most frequent words, and apply five rounds of Iterative Normalization (Zhang et al., 2019).

I use dictionaries from MUSE (Conneau et al., 2018a), a popular BLI benchmark, with standard splits: train on 5K source word translations and test on 1.5K words for BLI. For each language, I train three projection-based CLWE: canonical correlation analysis (Faruqui and Dyer, 2014, CCA), MSE+Orth using the implementation by Conneau et al. (2018a), and Relaxed CSLS loss (Joulin et al., 2018, RCSLS). I retrofit these CLWE to the training dictionary (pink in figures) and to both the training and the synthetic dictionary (orange in figures).

In MUSE, words from the training dictionary have higher frequencies than words from the

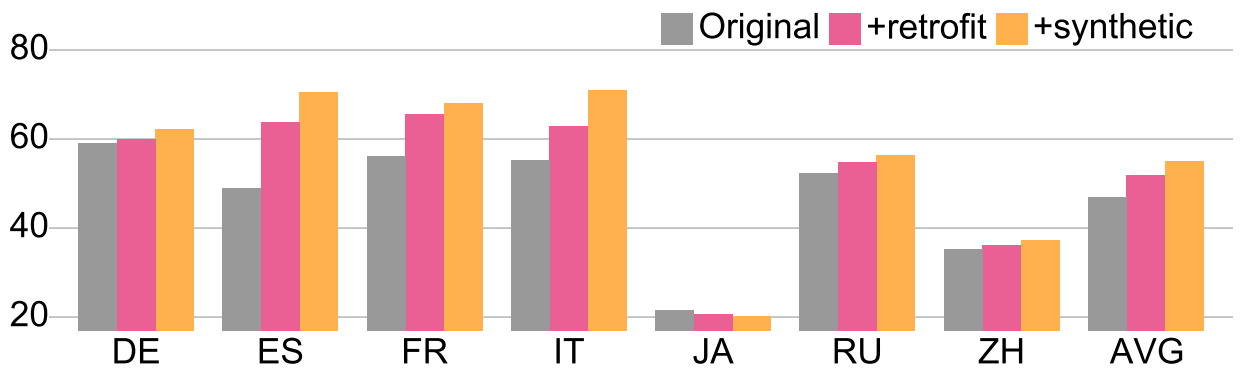


Figure 4.3: Dependency parsing with MSE+Orth

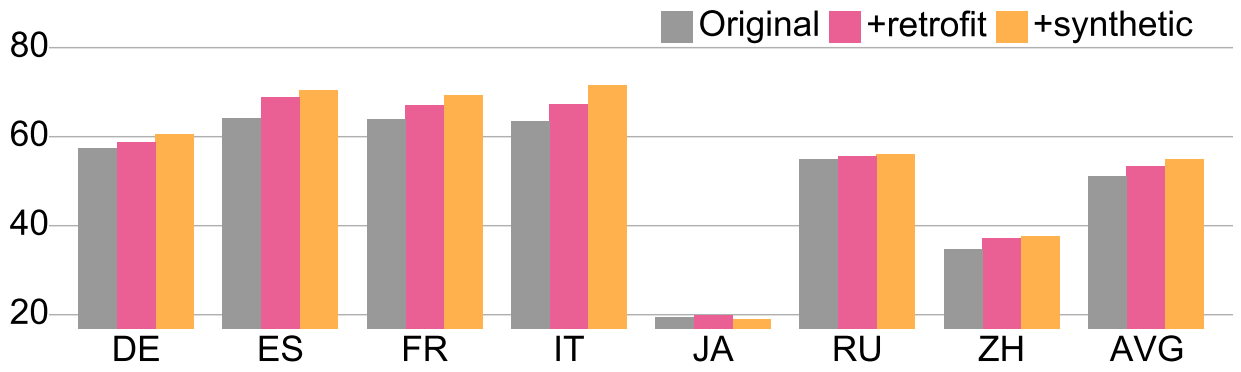


Figure 4.4: Dependency parsing with CCA

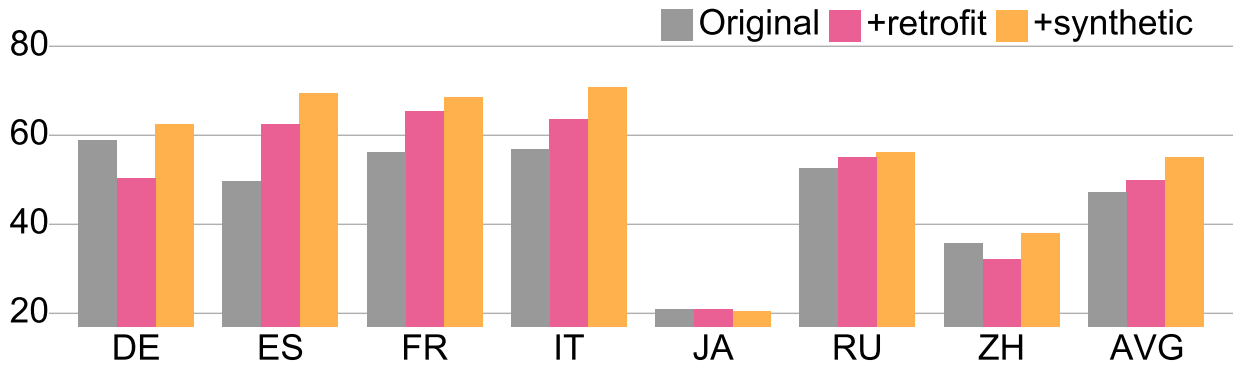


Figure 4.5: Dependency parsing with RCSLS

Figure 4.6: For each CLWE, we report accuracy for document classification (left) and unlabeled attachment score (UAS) for dependency parsing (right). Compared to the original embeddings (gray), retrofitting to the training dictionary (pink) improves average downstream task scores, confirming that fully exploiting the training dictionary helps downstream tasks. Adding a synthetic dictionary (orange) further improves test accuracy in some languages.

test set.[3] For example, the most frequent word in the English-French test dictionary is "torpedo", while the training dictionary has translations for frequent words such as "the" and "good". More frequent words are likely to be more salient in downstream tasks, so underfitting these more frequent training pairs hurts generalization to downstream tasks.[4]

4.3.4 Evaluation on Dependency Parsing

Multiple CLWE are compared on dependency parsing. When evaluating on dependency parsing, the embedding layer of the model to CLWE and use the zero-shot setting, where a model is trained in English and evaluated in the target language.

I test on dependency parsing, a structured prediction task. I use Universal Dependencies (Nivre et al., 2019a) v2.4 with the standard split. I use the biaffine parser (Dozat and Manning, 2017) in AllenNLP (Gardner et al., 2017) with the same hyperparameters as Ahmad et al. (2019a). To focus on the influence of CLWE, I remove part-of-speech features following Ammar et al. (2016). I report the average unlabeled attachment score (UAS) of five runs.

Results Although training dictionary retrofitting lowers BLI test accuracy, it improves both downstream tasks' test accuracy (Figure 4.6). This confirms that over-optimizing the test BLI accuracy can hurt downstream tasks because training dictionary words are also important. The synthetic dictionary further improves downstream models, showing that generalization to downstream tasks must balance between BLI training and test accuracy.

Qualitative Analysis As a qualitative example, coordinations improve after retrofitting to the training dictionary. For example, in the German sentence "Das Lokal ist sauber, hat einen gemütlichen 'Raucherraum' und wird gut besucht", the bar ("Das Lokal") has three properties: it is clean, has a smoking room, and is popular. However, without retrofitting, the final property "besucht" is connected to "hat" instead of "sauber"; i.e., the final clause stands on its own. After retrofitting to the English-German training dictionary, "besucht" is moved closer to its English

[3] `https://github.com/facebookresearch/MUSE/issues/24`

[4] A pilot study confirms that retrofitting to infrequent word pairs is less effective.

translation “visited” and is correctly parsed as a property of the bar.

4.4 Readability Estimation for Second Language Learners

In the prior sections, the focus has been common downstream NLP tasks such as document classification and dependency parsing. In this section, I now use cross-lingual embeddings for readability prediction task.

Accurately estimating the readability or difficulty of words and text has been an important fundamental task in NLP and education, with a wide range of applications including reading resource suggestion (Heilman et al., 2008), text simplification (Yimam et al., 2018), and automated essay scoring (Vajjala and Rama, 2018).

A number of linguistic resources have been created either manually or semi-automatically for non-native learners of languages such as English (Capel, 2010, 2012), French (François et al., 2014), and Swedish (François et al., 2016; Alfter and Volodina, 2018), often referencing the Common European Framework of Reference (Council of Europe, 2001, CEFR). However, few linguistic resources exist outside these major European languages and manually constructing such resources demands linguistic expertise and efforts.

This led to the proliferation of NLP-based *readability* or *difficulty assessment* methods to automatically estimate the difficulty of words (Alfter and Volodina, 2018) and texts (Vajjala and Meurers, 2012; Wang and Andersen, 2016; Vajjala and Rama, 2018; Settles et al., 2020). However, bootstrapping lexical resources with difficulty information often assumes the existence of textual datasets (e.g., digitized coursebooks) annotated with difficulty. Similarly, many text readability estimation methods (Wang and Andersen, 2016; Xia et al., 2016) assume the existence of abundant lexical or grammatical resources annotated with difficulty information. Individual research studies focus only on one side, either words or texts, although in reality they are closely intertwined—there is a *recursive relationship between word and text difficulty*, where the difficulty of a word is correlated to the *minimum* difficulty of the document where that word appears, and the difficulty of a document is correlated to the *maximum* difficulty of a word in that document (Figure 4.8).

I propose a method to jointly estimate word and text readability in a semi-supervised fashion

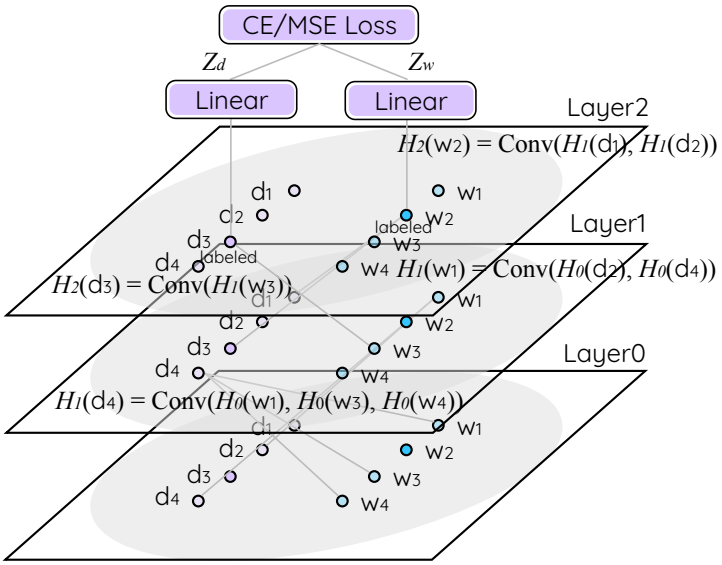


Figure 4.7: Overview of the proposed GCN architecture which recursively connects word w_i and document d_j to exploit the recursive relationship of their difficulty.

from a smaller number of labeled data by leveraging the recursive relationship between words and documents. Specifically, I leverage recent developments in graph convolutional networks (Kipf and Welling, 2017, GCNs) and predict the difficulty of words and documents simultaneously by modeling those as nodes in a graph structure and recursively inferring their embeddings using the convolutional layers (Figure 4.7). Our model leverages not only the supervision signals but also the recursive nature of word-document relationship. The contributions of this paper are two fold:

- I reframe the word and document readability estimation task as a semi-supervised, joint estimation problem motivated by their recursive relationship of difficulty.
- I show that GCNs are effective for solving this by exploiting unlabeled data effectively, even when less labeled data is available.

4.4.1 Task Definition

Given a set of words $\mathcal{W}$ and documents $\mathcal{D}$, the goal of the joint readability estimation task is to find a function f that maps both words and documents to their difficulty label f :

$\mathcal{W} \cup \mathcal{D} \rightarrow Y$. Documents here can be text of an arbitrary length, although I use paragraphs as the basic unit of prediction. This task can be solved as a classification problem or a regression problem where $Y \in \mathbb{R}$. I use six CEFR-labels representing six levels of difficulty, such as $Y \in$ {A1 (lowest), A2, B1, B2, C1, C2 (highest)} for classification, and a real-valued readability estimate $\beta \in \mathbb{R}$ inspired by the item response theory (Lord, 1980, IRT) for regression[5] . The β for each six CEFR level are A1= -1.38, A2= -0.67, B1= -0.21, B2= 0.21, C1= 0.67, and C2= 1.38.

Words and documents consist of mutually exclusive unlabeled subsets $\mathcal{W}_U$ and $\mathcal{D}_U$ and labeled subsets $\mathcal{W}_L$ and $\mathcal{D}_L$. The function f is inferred using the supervision signal from $\mathcal{W}_L$ and $\mathcal{D}_L$, and potentially other signals from $\mathcal{W}_U$ and $\mathcal{D}_U$ (e.g., relationship between words and documents).

4.4.2 Exploiting Recursive Relationship by Graph Convolutional Networks

I first show how the readability of words and documents are recursively related to each other. I then introduce a method based on graph convolutional networks (GCN) to capture such relationship.

4.4.2.1 Recursive Relationship of Word and Document Difficulty

The motivation of using a graph-based method for difficulty classification is the recursive relationship of word and document difficulty. Figure 4.8 shows such recursive relationship using the difficulty-labeled datasets explained in Section 5. One insight here is the strong correlation between the difficulty of a document and *the maximum difficulty of a word in that document*. This is intuitive and shares motivation with a method which exploits hierarchical structure of a document (Yang et al., 2016). However, the key insight here is the strong correlation between the difficulty of a word and *the minimum difficulty of a document where that word appears*, indicating that the readability of words informs that of documents, and vise versa.

[5] I assumed the difficulty estimate β is normally distributed and used the mid-point of six equal portions of $N(0, 1)$ when mapping CEFR levels to β.

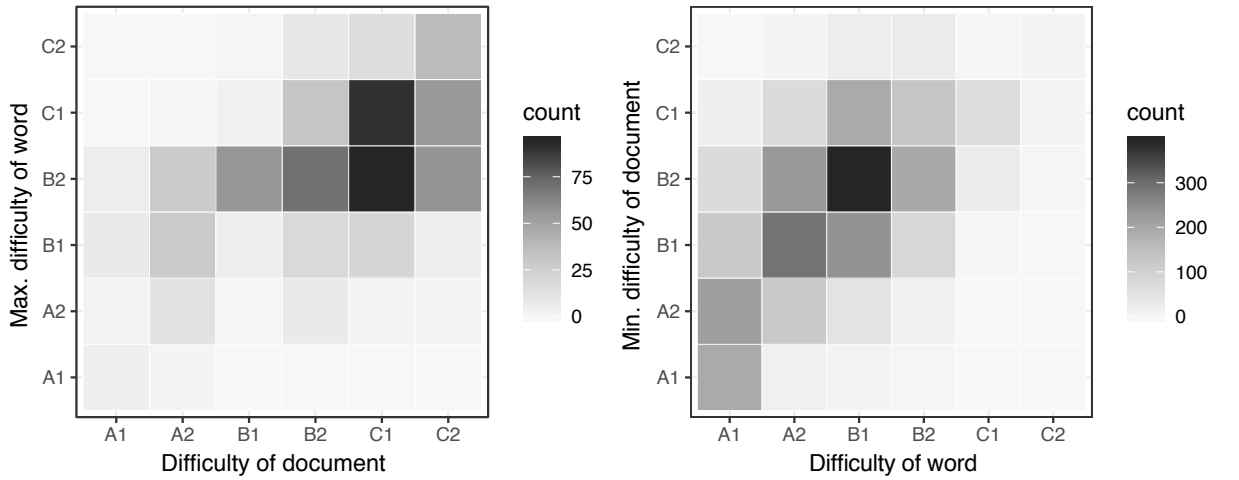


Figure 4.8: Recursive relationship of word/document difficulty. Word difficulty is correlated to the *minimum* difficulty of the document where that word appears, and document difficulty is correlated to the *maximum* difficulty of a word in that document.

4.4.2.2 Graph Convolutional Networks on Word-Document Graph

To capture the recursive, potentially nonlinear relationship between word and document readability while leveraging supervision signals and features, I propose to use graph convolutional networks (Kipf and Welling, 2017, GCNs) specifically built for text classification (Yao et al., 2019), which treats words and documents as nodes. Intuitively, the hidden layers in GCN, which recursively connects word and document nodes, encourage exploiting the recursive word-document relationship.

Given a heterogeneous word-document graph $G = (V, E)$ and its adjacency matrix $A \in \mathbb{R}^{|V|\times|V|}$, the hidden states for each layer $H_n \in \mathbb{R}^{|V|\times h_n}$ in a GCN with N hidden layers is computed using the previous layer H_{n-1} as:

$$H_n = \sigma(\tilde{A}H_{n-1}W_n) \tag{4.17}$$

where σ is the ReLU function[6] , $\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ i.e., a symmetrically normalized matrix of A with its degree matrix D, and $W_n \in \mathbb{R}^{h_{n-1}\times h_n}$ is the weight matrix for the nth layer. The input

[6] A simplified version of GCN with linear layers (Wu et al., 2019) in preliminary experiments shows that hidden layers with ReLU performed better.

to the first layer H_1 is $H_0 = X$ where $X \in \mathbb{R}^{|V| \times h_0}$ is the feature matrix with h_0 dimensions for each node in V. I use three different edge weights following Yao et al. (2019): (1) $A_{ij} = \text{tfidf}_{ij}$ if i is a document and j is a word, (2) the normalized point-wise mutual information (PMI) i.e., $A_{ij} = \text{PMI}(i, j)$ if both i and j are words, and (3) self-loops, i.e., $A_{ii} = 1$ for all i.

I now describe the components which differs from Yao et al. (2019). I use separate final linear layers for words and documents[7] :

$$Z_w = H_N W_w + b_w \tag{4.18}$$

$$Z_d = H_N W_d + b_d \tag{4.19}$$

where W and b are the weight and bias of the layer, and used a linear combination of word and document losses weighted by α (Figure 4.7)

$$\mathcal{L} = \alpha \mathcal{L}(Z_w) + (1 - \alpha)\mathcal{L}(Z_d) \tag{4.20}$$

For regression, I used Z (Z_w for words and Z_d for documents) as the prediction of node v and used the mean squared error (MSE):

$$\mathcal{L}(Z) = \frac{1}{|V_L|} \sum_{v \in V_L} (Z_v - Y_v)^2 \tag{4.21}$$

where $V_L = \mathcal{W}_L \cup \mathcal{D}_L$ is the set of labeled nodes. For classification, I use a softmax layer followed by a cross-entropy (CE) loss:

$$\mathcal{L}(Z) = -\sum_{v \in V_L} \log \frac{\exp(Z_{v,Y_v})}{\sum_i \exp(Z_{v,i})}. \tag{4.22}$$

Since GCN is transductive, node set V also includes the unlabeled nodes from the evaluation sets and have predicted difficulty labels assigned when training is finished.

4.4.3 Experiments

Datasets I use publicly available English CEFR-annotated resources for second language learners, such as CEFR-J (Negishi et al., 2013) Vocabulary Profile as words and Cambridge English

[7] A model variant with a common linear layer (i.e., original GCN) for both words and documents did not perform as well.

Dataset	Train	Dev	Test
Words (CEFR-J + C1/C2)	2,043	447	389
Documents (Cambridge + A1)	482	103	98

Table 4.4: Dataset size for words and documents

Readability Dataset (Xia et al., 2016) as documents (Table 4.4). Since these two datasets lack C1/C2-level words and A1 documents, I hired a linguistic PhD to write these missing portions[8] .

Baselines I compare our method against methods used in previous work (Feng et al., 2010; Vajjala and Meurers, 2012; Martinc et al., 2019; Deutsch et al., 2020): (1) logistic regression for classification (LR cls), (2) linear regression for regression (LR regr), (3) Gradient Boosted Decision Tree (GBDT), and (4) Hierarchical Attention Network (Yang et al., 2016, HAN), which is reported as one of the state-of-the-art methods in readability assessment for documents (Martinc et al., 2019; Deutsch et al., 2020).

Features For all methods except for HAN, I use both surface or "traditional" (Vajjala and Meurers, 2012) and embedding features on words and documents which are shown to be effective for readability estimation (Culligan, 2015; Settles et al., 2020; Deutsch et al., 2020). For words, I use their length (in characters), the log frequency in Wikipedia (Ginter et al., 2017), and GloVe (Pennington et al., 2014). For documents, I use the number of NLTK (Loper and Bird, 2002)-tokenized words in a document, and the output of embeddings from BERT-base model (Devlin et al., 2019) which are averaged over all tokens in a given sentence.

Hyperparameters I conduct random hyperparameter search with 200 samples, separately selecting two different sets of hyperparameters, one optimized for word difficulty and the other for document. I set the number of hidden layers $N = 2$ with $h_n = 512$ for documents and $N = 1$ with $h_n = 64$ for words.

Evaluation I use accuracy and Spearman's rank correlation as the metrics. When calculating the correlation for a classification model, I convert the discrete outputs into continuous values in two ways: (1) convert the CEFR label with the maximum probability into corresponding

[8] The dataset is available at `https://github.com/openlanguageprofiles/olp-en-cefrj`.

Method	Word Acc	Word Corr	Document Acc	Document Corr
HAN	-	-	0.367	0.498
LR (regr)	0.409	0.534	0.480	0.657
LR (cls+m)	0.440	0.514	0.765	0.723
LR (cls+w)	0.440	0.540	0.765	0.880
GBDT	0.432	0.376	0.765	0.833
GCN (regr)	0.434	0.579	0.643	0.849
GCN (cls+m)	**0.476**	0.536	**0.796**	0.878
GCN (cls+w)	**0.476**	**0.592**	**0.796**	**0.891**

Table 4.5: Difficulty estimation results in accuracy (Acc) and correlation (Corr) on classification outputs converted to continuous values by taking the max (cls+m) or weighted sum (cls+w) and regression (regr) variants for the logistic regression (LR) and GCN.

β in Section 4.4.1, (cls+m), or (2) take a sum of all β in six labels weighted by their probabilities (cls+w).

4.4.3.1 Results

Table 4.5 shows the test accuracy and correlation results. GCNs show increase in both document accuracy and word accuracy compared to the baseline. I infer that this is because GCN is good at capturing the relationship between words and documents. For example, the labeled training documents include an A1 document and that contains the word "bicycle," and the difficulty label of the document is explicitly propagated to the "bicycle" word node, whereas the logistic regression baseline mistakenly predicts as A2-level, since it relies solely on the input features to capture its similarities.

4.4.3.2 Ablation Study on Features

Table 4.6 shows the ablation study on the features explained in Section 4.4.3. By comparing Table 4.5 and Table 4.6, which are experimented on the same datasets, GCN without using any traditional or embedding features ("None") shows comparative results to some baselines, especially on word-level accuracy. Therefore, the structure of the word-document graph provides effective

	Word		Document	
Features	Acc	Corr	Acc	Corr
All	0.476	0.592	0.796	0.891
−word freq.	0.476	0.591	0.796	0.899
−doc length	0.481	0.601	0.796	0.890
−GloVe	0.463	0.545	0.714	0.878
−BERT	0.450	0.547	0.684	0.830
None	0.440	0.436	0.520	0.669

Table 4.6: Ablation study on the features used. "None" is when applying GCN without any features ($X = I$ i.e., one-hot encoding per node), which solely relies on the word-document structure of the graph.

and complementary signal for readability estimation.

Overall, the BERT embedding is a powerful feature for predicting document readability on Cambridge Readabilty Dataset. Ablating the BERT embeddings (Table 4.6) significantly decreases the document accuracy (-0.112) which is consistent with the previous work (Martinc et al., 2019; Deutsch et al., 2020) that BERT being one of the best-performing method for predicting document readability on one of the datasets they used, and HAN performing relatively low due to not using the BERT embeddings.

4.4.4 Training on Less Labeled Data

To analyze whether GCN is robust when training dataset is small, I compare the baseline and GCN by varying the amount of labeled training data. In Figure 4.9, I observe consistent improvement in GCN over the baseline especially in word accuracy. This outcome suggests that the performance of GCN stays robust even with smaller training data by exploiting the signals gained from the recursive word-document relationship and their structure. Another trend observed in Figure 4.9 is the larger gap in word accuracy compared to document accuracy when the training data is small likely due to GCN explicitly using context given by word-document edges.

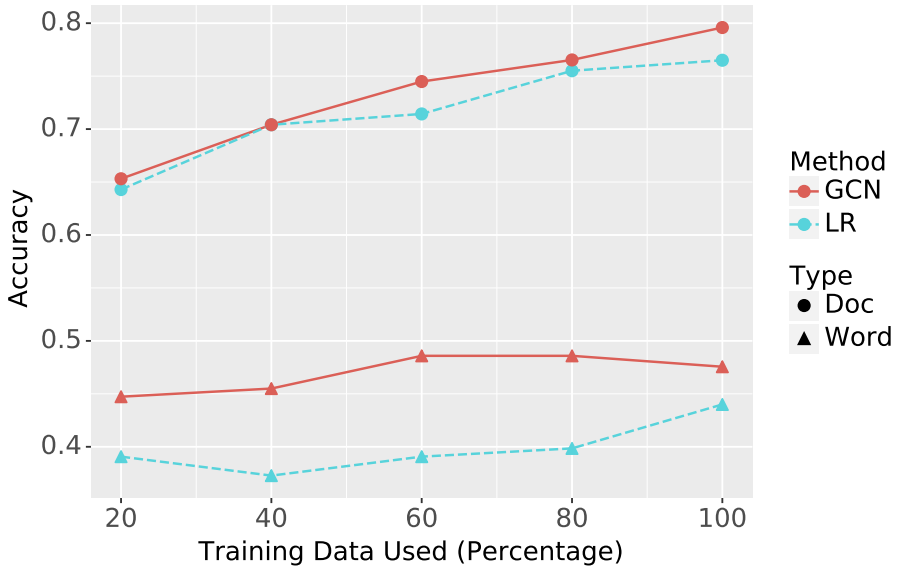


Figure 4.9: Word and document accuracy with different amount of training data used.

4.4.5 Cross-Lingual Transfer of Readability Estimation

I now turn to cross-lingual transfer of readability estimation task. Compared to cross-lingual document classification for identifying topic of documents, semantically similar words does not necessarily have the same labels i.e., same difficulty,

Experiment setup I follow the zero-shot cross-lingual setting, i.e., not using English data during training. Instead of using BERT and GloVe embeddings, mBERT and cross-lingual word embeddings by Bojanowski et al. (2017) and Joulin et al. (2018) are used. For the English data, the same data as Section 4.4 are used and For the Swedish data, the COCTAIL dataset (Volodina and Kokkinakis, 2012) are used.

Results Table 4.7 shows the results of the cross-lingual difficulty estimation task, training on English and evaluating on Swedish. The recursive relationship of word and document difficulties and further conducting joint training of word and document level are beneficial in the cross-lingual setting too. Following the same trend as the monolingual setting, there is larger improvement in word-level estimation than the document-level estimation.

	Word		Document	
Method	Acc	Corr	Acc	Corr
LR (regr)	0.218	-0.067	0.032	-0.012
LR (cls+m)	0.219	-0.124	0.346	0.126
LR (cls+w)	0.219	-0.091	0.346	0.217
GBDT	0.229	-0.075	0.340	0.075
GCN (regr)	0.132	-0.080	0.008	0.093
GCN (cls+m)	0.245	0.357	0.392	0.137
GCN (cls+w)	0.245	0.477	0.392	0.209

Table 4.7: Cross-lingual difficulty estimation results in accuracy (Acc) and correlation (Corr) on classification outputs converted to continuous values by taking the max (cls+m) or weighted sum (cls+w) and regression (regr) variants for the logistic regression (LR) and GCN.

This chapter focuses on directly modifying the objective function to train or post-process cross-lingual word embeddings. We show that cross-lingual transfer on downstream tasks improves by modifying objective functions, but the magnitude of improvement and which method to use largely depends on a downstream task. In this dissertation so far, I focus on evaluating and improving cross-lingual word embeddings during. In the next chapter, I turn to focusing on a different family of models, namely pretrained language models, and analyze the effect of using different languages during pretraining.

Chapter 5

Analysis on Multilingual Pretraining of Transformers

5.1 Introduction

Pretrained multilingual language models Devlin et al. (2019); Conneau et al. (2020) are now a standard approach for cross-lingual transfer in natural language processing (NLP). However, there are multiple, potentially related issues on pretraining multilingual models. Conneau et al. (2020) find "curse of multilinguality": for a fixed model size, zero-shot performance on target languages seen during pretraining increases with additional pretraining languages only until a certain point, after which performance decreases. Wang et al. (2020b) also report "negative interference", where monolingual models achieve better results than multilingual models, both on subset of high- and low-resource languages. However, those findings are limited to target languages seen during pretraining.

Current multilingual models cover only a small subset of the world's languages. Furthermore, due to data sparsity, monolingual pretrained models are not likely to obtain good results for many low-resource languages, such as dialects or extinct languages. In those cases, multilingual models can zero-shot learn for unseen languages with an above-chance performance, which can be further improved via model adaptation of target-language text (Wang et al., 2020a) even with limited amounts (Ebrahimi and Kann, 2021). However, it is poorly understood how the number of pretraining languages influences performance in those cases. Does the "curse of multilinguality" or the "negative interference" also negatively impact performance on unseen target languages? And, if I want a model to be applicable to as many unseen languages as possible, how many languages

should it be trained on?

Specifically, I ask the following research questions: (1) How does pretraining on an increasing number of languages impact zero-shot performance on unseen target languages? (2) Does the effect of the number of pretraining languages change with model adaptation to target languages? (3) Does the answer to the first research question change if the pretraining languages are all related to each other?

I pretrain a variety of monolingual and multilingual models, which I then finetune on English and apply to three zero-shot cross-lingual downstream tasks in unseen target languages: part-of-speech (POS) tagging, named entity recognition (NER), and natural language inference (NLI). Without model adaptation, increasing the number of pretraining languages improves accuracy on unrelated unseen target languages at first and plateaus thereafter. With adaptation, additional pretraining languages beyond English generally help. Last, choosing a diverse set of pretraining languages is crucial for effective transfer.

Due to the intense computational cost of pretraining and its environmental impact (Strubell et al., 2019), we opt to perform our experiments in Section 5.4 on a relatively small scale with a fixed computational budget for each model, but validate our most central findings in Section 5.5 on large pretrained models which are already publicly available.

5.2 Cross-lingual Transfer via Pretraining

Pretrained multilingual models are a straightforward cross-lingual transfer approach: a model pretrained on multiple languages is then fine-tuned on target-task data in the **source** language. Subsequently, the model is applied to target-task data in the **target** language. Most commonly, the target language is part of the model's pretraining data. However, cross-lingual transfer is possible even if this is not the case, though performance tends to be lower. This paper extends prior work exploring the cross-lingual transfer abilities of pretrained models for **seen** target languages depending on the number of pretraining languages to **unseen** target languages. I now transfer via pretrained multilingual models and introduce the models and methods vetted in our experiments.

5.2.1 Background and Methods

Pretrained Language Models Contextual representations such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) are not just useful for monolingual representations. Multilingual BERT (Devlin et al., 2019, mBERT), XLM (Lample and Conneau, 2019), and XLM-RoBERTa (Conneau et al., 2020, XLM-R) have surprisingly high cross-lingual transfer performance compared to the previous best practice: static cross-lingual word embeddings (Pires et al., 2019; Wu and Dredze, 2019). Multilingual models are also practical—why have hundreds of separate models for each language when you could do better with just one? However, there's a catch. Conneau et al. (2020) reports the "curse of multilinguality": as the number of pretraining languages increases, the better the cross-lingual transfer is up to some point, after which the performance in zero-shot transfer starts to decrease. Our work complements Conneau et al. (2020) by explicitly dividing up between seen and unseen languages during pretraining and testing whether the "curse of multilinguality" is also an issue for unseen languages too.

Model Adaptation to Unseen Languages Adapting pretrained multilingual models, such as mBERT and XLM-R, to unseen languages is one way to use such models beyond the languages covered during pretraining time. Several methods for adapting pretrained multilingual language models to unseen languages have been proposed in the literature, including continuing masked language model (MLM) training (Chau et al., 2020; Müller et al., 2020) optionally adding Adapter modules (Pfeiffer et al., 2020), or extending the vocabulary of the pretrained models (Artetxe et al., 2020; Wang et al., 2020a). However, such adaptation methods assume the existence of sufficient monolingual corpora in the target languages. Some spoken languages, dialects, or extinct languages lack monolingual corpora to conduct model adaptation, which motivates us to look into the performance of languages unseen during pretraining.

5.2.2 Research Questions

A single pretrained model that can be applied to any language, including those unseen during pretraining, is both more efficient and more practical than pretraining one model per language. Moreover, it is the only practical option for unknown target languages or for languages without enough resources for pretraining. Thus, models that can be applied or at least easily adapted to unseen languages are an important research focus. This work addresses the following research questions (RQ), using English as the source language for finetuning.

RQ1*: How does the number of pretraining languages influence zero-shot performance on unseen target languages?*

I first explore how many languages a model should be pretrained on if the target language is unknown at test time or has too limited monolingual resources for model adaptation. On one hand, I hypothesize that increasing the number of pretraining languages will improve performance, as the model sees a more diverse set of scripts and linguistic phenomena. Also, the more pretraining languages, the better chance of having a related language to the target language. However, multilingual training can cause interference: other languages could distract from English, the finetuning source language, and thus, lower performance.

RQ2*: How does the answer to RQ1 change with model adaptation to the target language?*

This question is concerned with settings in which I have enough monolingual data available to adapt a pretrained model to the target language. Similar to our hypothesis for RQ1, I expect that having seen more pretraining languages should make adaptation to unseen target languages easier. However, another possibility is that the process of adapting the model makes any languages other than the finetuning source language unnecessary, such that performance stays the same or decreases when adding more pretraining languages.

RQ3*: Do the answers to RQ1 change if all pretraining languages are related to each other?*

I use a diverse set of pretraining languages when exploring RQ1, since I expect that to be maximally beneficial. However, the results might change depending on the exact languages. Thus,

as a case study, I repeat all experiments using a set of closely related languages. On the one hand, I hypothesize that benefits due to adding more pretraining languages (if any) will be smaller with related languages, as I reduce the diversity of linguistic phenomena in the pretraining data. However, on the other hand, if English is all I use during fine-tuning, performance might increase with related languages, as this will approximate training on more English data more closely.

5.3 Experimental Setup

Pretraining Corpora All our models are pretrained on the CoNLL 2017 Wikipedia dump (Ginter et al., 2017). To use equal amounts of data for all pretraining languages, we downsample all Wikipedia datasets to have equal number of sequences. I standardize to the smallest corpus, Hindi. The resulting pretraining corpus size is around 200MB per language.[1] I hold out 1K sequences after preprocessed to be around 512 tokens per sequence as a development set to track the models' performance during pretraining.

Corpora for Model Adaptation For model adaptation (RQ2), I select unseen target languages contained in both XNLI (Conneau et al., 2018b) and Universal Dependencies 2.5 (Nivre et al., 2019b): Farsi (FA), Hebrew (HE), French (FR), Vietnamese (VI), Tamil (TA), and Bulgarian (BG). Model adaptation is typically done for low-resource languages not seen during pretraining because monolingual corpora are too small (Wang et al., 2020a). Therefore, I use the Johns Hopkins Bibles corpus by McCarthy et al. (2020) following Ebrahimi and Kann (2021).[2]

Tasks I evaluate our pretrained models on the following downstream tasks from the XTREME dataset (Hu et al., 2020): POS tagging and NLI. For the former, I select 29 languages from Universal Dependencies v2.5 (Nivre et al., 2019b). For the latter, I use all fifteen languages in XNLI (Conneau et al., 2018b). I follow the default train, validation, and test split in the XTREME dataset.

Models and Hyperparameters Following Conneau et al. (2020)'s XLM-R Base model,

[1] Micheli et al. (2020) show that corpora of at least 100MB are reasonable for pretraining.
[2] In cases where multiple versions of the Bible are available in the target language, I select the largest one.

Langs	Tasks
Seen languages	
English (EN)	POS, NER, NLI
Russian (RU)	POS, NER, NLI
Arabic (AR)	POS, NER, NLI
Chinese (ZH)	POS, NER, NLI
Hindi (HI)	POS, NER, NLI
Spanish (ES)	POS, NER, NLI
Greek (EL)	POS, NER, NLI
Finnish (FI)	POS, NER
Indonesian (ID)	POS, NER
Turkish (TR)	POS, NER, NLI
German (DE)	POS, NER, NLI
Dutch (NL)	POS, NER, NLI
Swedish (SV)	-
Danish (DA)	-
Unseen languages	
Bulgarian (BG)	POS, NER, NLI
French (FR)	POS, NER, NLI
Urdu (UR)	POS, NER, NLI
Africaans (AF)	POS, NER
Basque (EL)	POS, NER
Estonian (ET)	POS, NER
Farsi (FA)	POS, NER
Hebrew (HE)	POS, NER
Hungarian (HU)	POS, NER
Italian (IT)	POS, NER
Japanese (JA)	POS, NER
Korean (KO)	POS, NER
Marathi (MR)	POS, NER
Portuguese (PT)	POS, NER
Vietnamese (VI)	POS, NER
Tamil (TA)	POS, NER
Telugu (TE)	POS, NER
Swahili (SW)	NLI
Thai (TH)	NLI

Table 5.1: Languages used in our experiments, aiming to cover languages from diverse language families.

I train transformers Vaswani et al. (2017) with 12 layers, 768 units, 12 attention heads, and a maximum of 512 tokens per sequence. To accommodate all languages and facilitate comparability between all pretraining setups, I use XLM-R's vocabulary and SentencePiece (Kudo and Richardson, 2018) tokenizer by Conneau et al. (2020).

I use masked language modeling (MLM) as our pretraining objective and, like Devlin et al.

Model	Pretraining Languages
Div-2	EN, RU
Div-3	EN, RU, ZH
Div-4	EN, RU, ZH, AR
Div-5	EN, RU, ZH, AR, HI
Div-6	EN, RU, ZH, AR, HI, ES
Div-7	EN, RU, ZH, AR, HI, ES, EL
Div-8	EN, RU, ZH, AR, HI, ES, EL, FI
Div-9	EN, RU, ZH, AR, HI, ES, EL, FI, ID
Div-10	EN, RU, ZH, AR, HI, ES, EL, FI, ID, TR
Rel-2	EN, DE
Rel-3	EN, DE, SV
Rel-4	EN, DE, SV, NL
Rel-5	EN, DE, SV, NL, DA

Table 5.2: Pretraining languages used for the models in our experiments: models are trained on a diverse set (Div-X) and related pretraining languages (Rel-X), with different numbers of pretraining languages.

(2019), mask 15% of the tokens. I pretrain all models for 150K steps, using Adam W (Loshchilov and Hutter, 2019) with a learning rate of 1×10^{-4} and a batch size of two on either NVIDIA RTX2080Ti or GTX1080Ti 12GB which approximately took four days to train each model. When pretraining, I preprocess sentences together to generate sequences of approximately 512 tokens. For continued pretraining, I use a learning rate of 2×10^{-5} and train for forty epochs, otherwise following the setup for pretraining. For finetuning, I use a learning rate of 2×10^{-5} and train for an additional ten epochs for POS tagging and an additional five epochs for NLI, following Hu et al. (2020).

Languages Table 5.1 shows languages used in our experiments. English is part of the pretraining data of all models. It is also the finetuning source language for all tasks, following Hu et al. (2020). I introduce two different sets for pretraining languages: "Diverse (Div)" and "Related (Rel)" (Table 5.2). I mainly focus on pretraining languages up to five, except for POS tagging where the trend is not clear and further experiment up to ten.

For POS tagging and NER, I regard 17 of the 29 languages available in XTREME as *unseen*, while the remaining 12 languages are pretraining languages for at least one model. For NLI, six languages are *seen* and the rest are *unseen*. The order in which I add pretraining languages follows

the size of their original CoNLL 2017 Wikipedia dumps, with larger sizes being added first.

5.4 Results

I now show the experimental results to investigate on each RQ.

5.4.1 Findings for RQ1

POS Tagging Figure 5.1 shows the POS tagging accuracy on the ten languages seen during pretraining and averaged over the 17 unseen languages. On average, models pretrained on multiple languages have higher accuracy on unseen languages than the model pretrained exclusively on English, showing that the model benefits from a more diverse set of pretraining data. However, the average accuracy only increases up to six languages. This indicates that our initial hypothesis "the more languages the better" might not be true.

Pretrain	en	ru	zh	ar	hi	bg	de	el	es	fr	sw	th	tr	ur	vi
EN	.731	.343	.340	.339	.345	.347	.375	.346	.404	.381	.366	.350	.358	.347	.354
Div-2	.725	.457	.336	.341	.342	.384	.373	.346	.421	.382	.364	.342	.354	.338	.352
Div-3	.738	.500	.485	.336	.338	.389	.374	.341	.412	.382	.354	.340	.345	.339	.345
Div-4	.718	.452	.467	.460	.350	.418	.398	.352	.439	.417	.379	.351	.369	.361	.361
Div-5	.717	.466	.484	.460	.462	.426	.382	.346	.443	.386	.370	.348	.356	.349	.349

Table 5.3: NLI accuracy on diverse pretraining languages over five seen (EN,RU,ZH,AR,HI) and 10 unseen languages. More pretraining languages up to five are also generally better for unseen languages in the NLI task.

Figure 5.2 provides a more detailed picture, showing the accuracy for different number of pretraining languages for all seen and unseen target languages. As expected, accuracy jumps when a language itself is added as a pretraining language. Furthermore, accuracy rises if a pretraining language from the same language family as a target language is added: for example, the accuracy of Marathi goes up by 9.3% after adding Hindi during pretraining, and the accuracy of Bulgarian increases by 31.2% after adding Russian. This shows that related languages are indeed beneficial for transfer learning. Also, (partially) sharing the same script with a pretraining language (e.g., ES and ET, AR and FA) helps with zero-shot cross-lingual transfer even for languages which are not

from the same family. These results are consistent with the outcome of Müller et al. (2020) and partially support the hypothesis by Pires et al. (2019) that shared scripts are effective on unseen languages.

With the question of how many pretraining languages are optimal, I obtain mixed results. One consistent finding is that, for the large majority of languages, using only English yields the worst results for unseen languages. However, as also shown by the average, increasing pretraining language does not necessarily improve the accuracy. This indicates that, while I want more than one pretraining language, using a smaller number than the 100 commonly used pretraining languages is likely sufficient unless I expect them to be closely related to one of the potential target languages. I further investigate it in Section 5.5.

NER Our NER results show a similar trend, and therefore, only report the average (Figure 5.3) and full details are available in Appendix **??**. For NER, the transfer to unseen languages are more limited, likely due to small subset of tokens are labeled as entities when compared to POS tags.

NLI Our NLI results show a similar trend (Table 5.3): accuracy on unseen languages plateaus at relatively small number of pretraining languages. Specifically, Div-4 has the highest accuracy for 8 target languages, while Div-5 is best only for two target languages. I again observe a gain in accuracy when adding related languages, such as an improvement of +3.7% accuracy for Bulgarian after adding Russian as a pretraining language.

5.4.2 Findings for RQ2

POS Tagging Figure 5.4 shows the POS tagging results for six languages after adaptation of the pretrained models via continued pretraining. As expected, accuracy is overall higher than in Figure 5.2. Importantly, there are accuracy gains in Farsi when adding Turkish (+9.8%) and in Hebrew when adding Greek (+7.7%), which are not observed before adapting models. I further investigate it in Section 5.5.

NER Figure 5.5 shows the results on the NER task. There are similarities between POS

tagging in the improvement of BG after adding RU. However, for example, there's limited improvement on FA even after adding AR regardless of partially shared scripts between the two languages. This indicates that the effect of adding related pretraining languages is task-dependent.

NLI For NLI, I see a small increase in accuracy after adding a second pretraining language. Results for two to five pretraining languages are similar for all target languages and, for Greek and Turkish, still similar to the English-only model. This indicates that, similar to our findings for POS tagging, a few pretraining languages could be sufficient for model adaptation. Finally, our NLI results are low overall. This is likely due to the size of the pretraining corpus is one of the top correlated feature for NLI (Lauscher et al., 2020) unlike POS tagging (Hu et al., 2020).

5.4.3 Findings for RQ3

Pretrain	en	de	ru	zh	ar	hi	bg	el	es	fr	sw	th	tr	ur	vi
EN	.731	.375	.343	.340	.339	.345	.347	.346	.404	.381	.366	.350	.358	.347	.354
Rel-2	.733	.536	.363	.350	.357	.361	.359	.367	.422	.384	.374	.360	.381	.363	.369
Rel-3	.721	.535	.351	.349	.350	.355	.350	.352	.434	.420	.383	.357	.382	.348	.370
Rel-4	.710	.493	.350	.336	.348	.355	.354	.349	.433	.409	.368	.360	.373	.347	.363
Rel-5	.726	.527	.339	.335	.335	.342	.343	.342	.430	.415	.376	.339	.372	.335	.347

Table 5.4: NLI accuracy on the 13 unseen languages using the models pretrained on related languages (EN, DE, SV, NL, DA), incrementally added one language at a time up to five languages.

POS Tagging In contrast to RQ1, POS tagging accuracy for most languages does not change much when increasing the number of pretraining languages (Figure 5.6). The unseen languages on which I observe gains belong to the Germanic (AF, NL), Romance (ES, IT, FR, PT), and Uralic (ET, FI, HU) language families. Those languages are spoken in Europe and relatively (as compared to the other language families) close to English. The accuracy on languages from other language families changes by $< 10\%$, which is smaller than the change for a diverse set of pretraining languages. This indicates that the models pretrained on similar languages struggle to transferring to unrelated languages.

NER F1 scores of EN, Rel-2, Rel-3, Rel-4, and Rel-5 are .218, .219, .227, .236, and .237 respectively. Compared to Div-X, pretraining on related languages also improves up to adding five

languages, however, those models brings relatively less improvement, similar to the POS tagging.

NLI Table 5.4 shows a similar trend for NLI: when adding related pretraining languages, accuracy on languages far from English either does not change much or decreases. In fact, for nine out of thirteen unseen target languages, Rel-5 is the worst.

5.5 More Pretraining Languages and Applying to Low-Resource Languages

Our main takeaways from the last section are: (RQ1) without model adaptation, increasing pretraining languages does not improve accuracy on unrelated unseen target languages; (RQ2) model adaptation largely helps exploiting models pretrained on more languages; and (RQ3) if using more than one pretraining language, diversity is important.

However, there are limitations in the experiment settings in Section 5.4. I assume the following: (1) relatively small pretraining corpora; (2) the target languages are included when building the model's vocabulary; (3) fixed computational resources; and (4) pretraining languages up to 10. I now explore if our findings in RQ1 and RQ2 hold without such limitations. For this, I use two publicly available pretrained XLM models (Lample and Conneau, 2019), which have been pretrained on full size Wikipedia in 17 (XLM-17) and 100 (XLM-100) languages, and XLM-R base model trained on a larger Common Crawl corpus (Conneau et al., 2020). I conduct a case study on low-resource languages unseen for all models, including unseen vocabularies: Maltese (MT), Wolof (WO), Yoruba (YO), Erzya (MYV), and Northern Sami (SME). All pretraining languages used in Div-X are included in XLM-17 except for Finnish, and all 17 pretraining languages for XLM-17 are a subset of the pretraining languages for XLM-100. I report the averages with standard deviations from three random seeds.

5.5.1 Results

RQ1 For models without adaptation, accuracy does not improve for increasing numbers of source languages (Table 5.5). Indeed, the accuracy on both XLM-17 and XLM-100 are on par even though the former uses 17 pretraining languages and the latter uses 100. One exception is on

Northern Sami (Uralic language with Latin script) due to XLM-17 not seeing any Uralic languages, but XLM-100 does during pretraining.

When further comparing Div-10 and XLM-17, addition of pretraining languages help for Maltese and Yoruba. Addition of Spanish even improved accuracy outside Romance languages, such as Maltese (Afro-Asiatic), and Wolof (Niger-Congo). Erzya remains constant from five to 100 languages, even increasing the pretraining corpus size from downsampled (Div-X) to full Wikipedia (XLM-17 and XLM-100), suggesting that the transfer across language families are primarily happening in languages written in Latin scripts. The pretraining corpus size is not an issue for POS tagging since Div-X models have comparable accuracy to XLM-17 and XLM-100 without the model adaptation.

RQ2 For the models with adaptation, there is a significant gap between XLM-17 and XLM-100. This confirms our findings in the last section: more pretraining languages may be beneficial. Thus, a possible explanation is that one or more of XLM-100's pretraining languages is similar to our target languages and such languages can be exploited through continued pretraining. Therefore, having the model see more languages during pretraining is better when the models can be adapted to each target language.

5.6 Related Work

Static Cross-Lingual Word Embeddings Static cross-lingual word embeddings (Mikolov et al., 2013a; Conneau et al., 2018a) embed and align words from multiple languages for downstream NLP tasks (Lample et al., 2018; Gu et al., 2018). However, since prior work (Ahmad et al., 2019b; Artetxe et al., 2020) report that pretrained multilingual language models are superior on languages seen during pretraining, most work has transitioned from static cross-lingual word embeddings to multilingual language models.

Analysis of Pretrained Multilingual Models on Seen Languages Starting from Pires et al. (2019), analysis of the cross-lingual transferability of pretrained multilingual language models has been a topic of interest. Pires et al. (2019) hypothesize that the cross-lingual transfer

occurs due to shared tokens across languages, but Artetxe et al. (2020) showed that cross-lingual transfer can be successful even among languages without shared scripts. Other work investigated the relationship between zero-shot cross-lingual learning and typological features (Lauscher et al., 2020), encoding language-specific features (Libovický et al., 2020), and mBERT's multlinguality (Dufter and Schütze, 2020). However, the majority of the analyses have either been limited to large public models (mBERT or XLM-R) or to up to two pretraining languages (K et al., 2020; Wu and Dredze, 2020). Here, I analyze the ability of models to benefit from an increasing number of pretraining languages.

5.7 Conclusion

This chapter explores pretraining on different numbers of languages has on unseen target languages after finetuning on English: (1) without model adaptation, improvement by pretraining on more languages is limited on unrelated unseen target languages; (2) with model adaptation, additional pretraining languages generally helps; and (3) when pretraining on more than one language, choosing a diverse set is important.

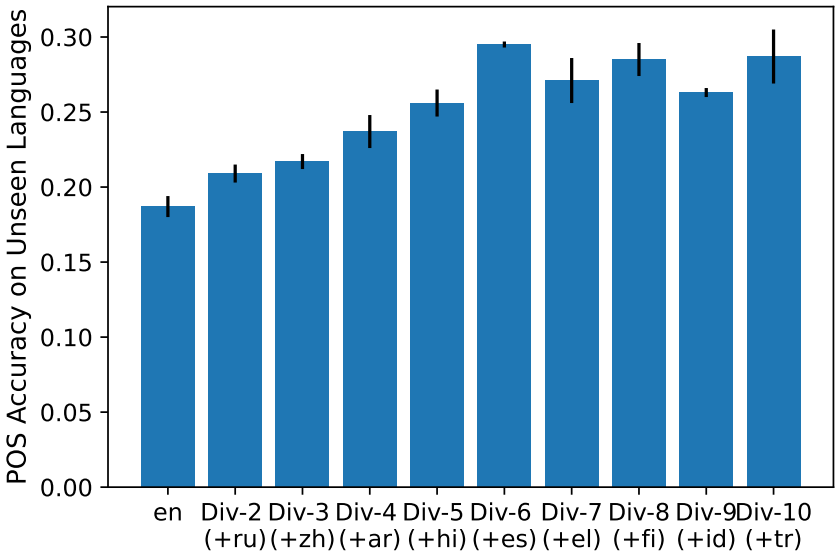


Figure 5.1: POS tagging accuracy after pretraining on a diverse set of up to 10 languages and finetuning on English. The accuracy improves until six languages on the given target languages.

Model	# lgs	MT	WO	YO	MYV	SME
Before Model Adaptation						
Div-5 (+hi)	5	.159 ± .019	.214 ± .029	.183 ± .012	.419 ± .016	.200 ± .012
Div-6 (+es)	6	.196 ± .021	.256 ± .020	.183 ± .004	.437 ± .010	.249 ± .001
Div-8 (+fi)	8	.176 ± .002	.236 ± .013	.178 ± .001	.435 ± .025	.320 ± .023
Div-10 (+tr)	10	.171 ± .004	.243 ± .016	.184 ± .007	.420 ± .008	.315 ± .013
XLM-17	17	.214 ± .007	.276 ± .009	.247 ± .033	.436 ± .035	.318 ± .010
XLM-100	100	.233 ± .017	.280 ± .019	.231 ± .026	.434 ± .027	.351 ± .026
XLM-R	100	.221 ± .033	.260 ± .030	.263 ± .034	.490 ± .022	.344 ± .052
After Model Adaptation						
Div-5 (+hi)	5	.272 ± .007	.388 ± .018	.251 ± .025	.489 ± .013	.361 ± .051
Div-6 (+es)	6	.310 ± .007	.366 ± .010	.253 ± .008	.492 ± .012	.377 ± .005
Div-8 (+fi)	8	.315 ± .002	.384 ± .036	.257 ± .017	.527 ± .016	.449 ± .005
Div-10 (+tr)	10	.298 ± .044	.389 ± .059	.251 ± .004	.508 ± .040	.457 ± .005
XLM-17	17	.451 ± .002	.410 ± .047	.361 ± .032	.603 ± .011	.567 ± .042
XLM-100	100	.627 ± .026	.673 ± .015	.385 ± .071	.697 ± .015	.752 ± .006
XLM-R	100	.594 ± .010	.689 ± .005	.509 ± .008	.720 ± .007	.703 ± .007

Table 5.5: POS tagging accuracy of our models pretrained on diverse languages, XLM-17, XLM-100, and XLM-R after finetuning on English. The models before adaptation are roughly on par regardless of the number of pretraining languages, and the models after adaptation is more affected by the related pretraining languages.

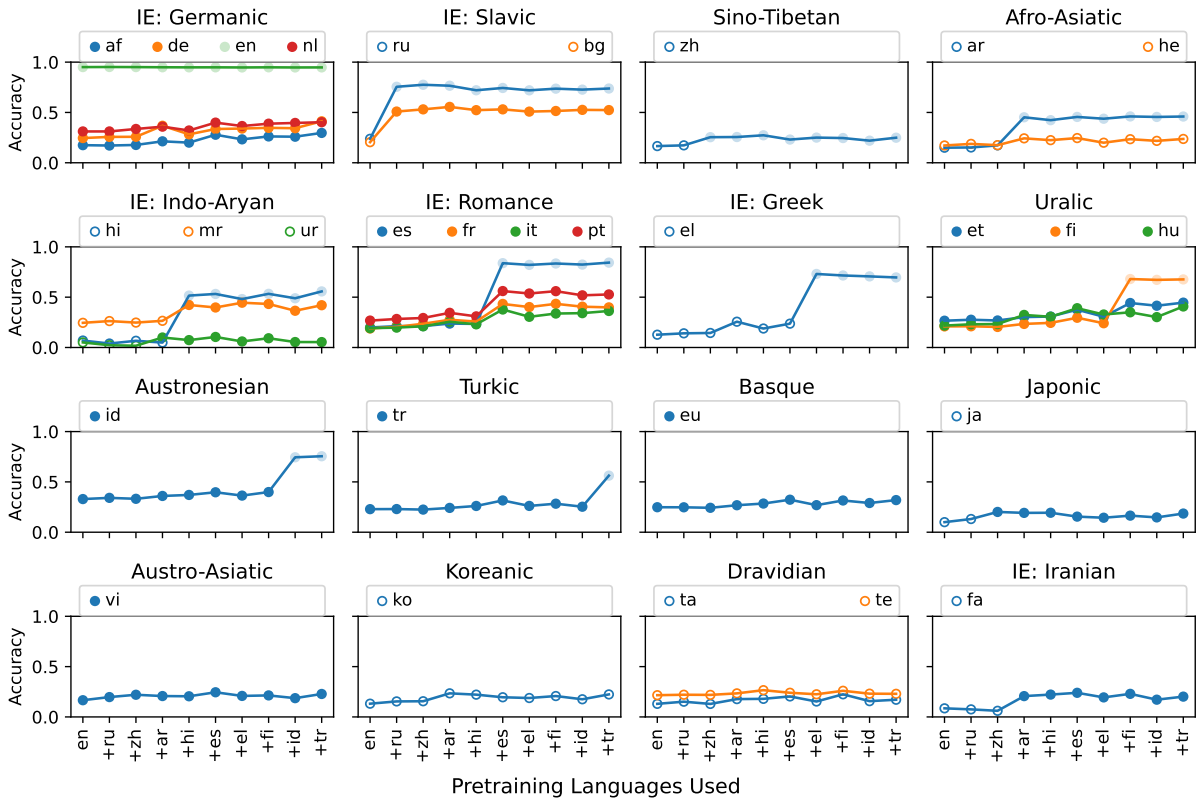


Figure 5.2: POS tagging accuracy on diverse pretraining languages (EN, RU, ZH, AR, HI, ES, EL, FI, ID, TR) grouped by families of target languages, with Indo-European (IE) languages further divided into subgroups following XTREME. Points in the plot are more transparent if the languages are seen during pretraining and filled circles represent when the script type of the target languages are common with one of the pretraining languages. The accuracy gain is significant for seen pretraining languages, and also the languages from the same family and the same script type of the pretraining languages when added.

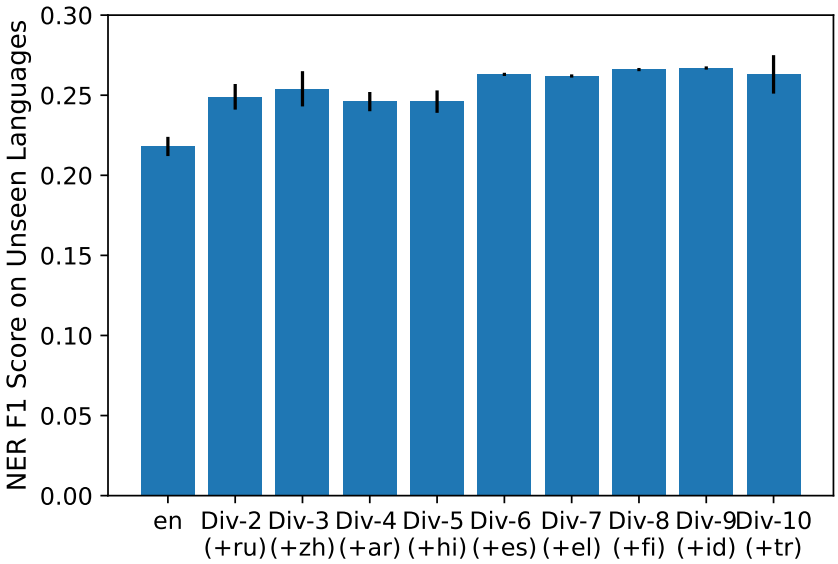


Figure 5.3: NER F1 score after pretraining on a diverse set of up to 10 languages and finetuning on English.

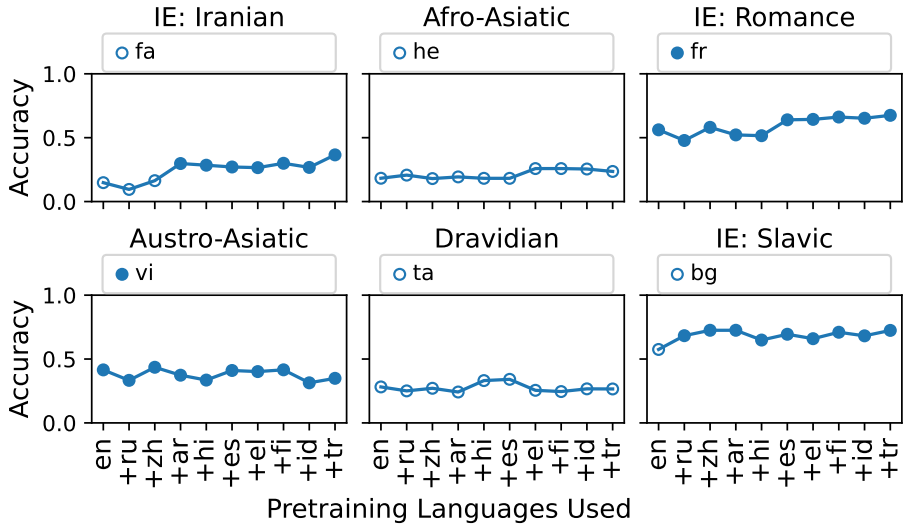


Figure 5.4: POS tagging accuracy after continued training on the Bible of each target language. The continued training gives surprising improvement in the model pretrained on multiple languages.

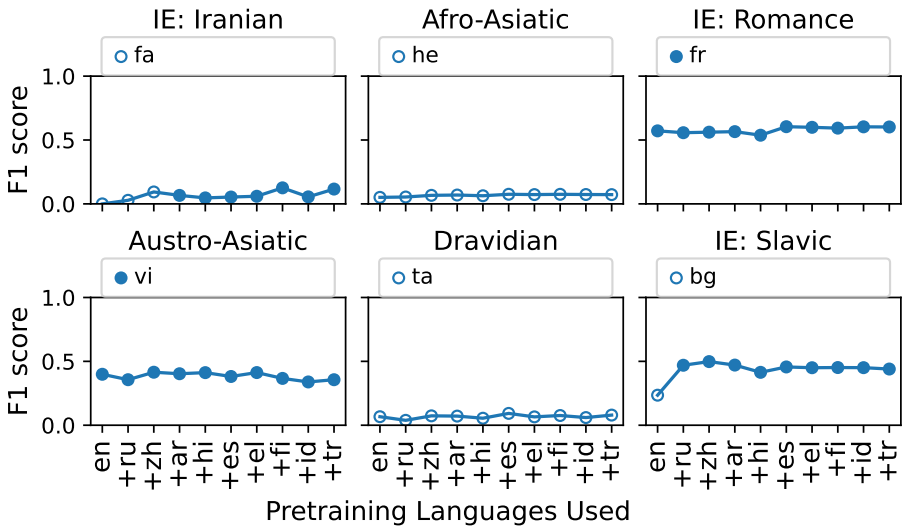


Figure 5.5: NER F1 scores after continued training on the Bible of each target language. The continued training gives limited improvement in most languages on NER when compared to POS tagging.

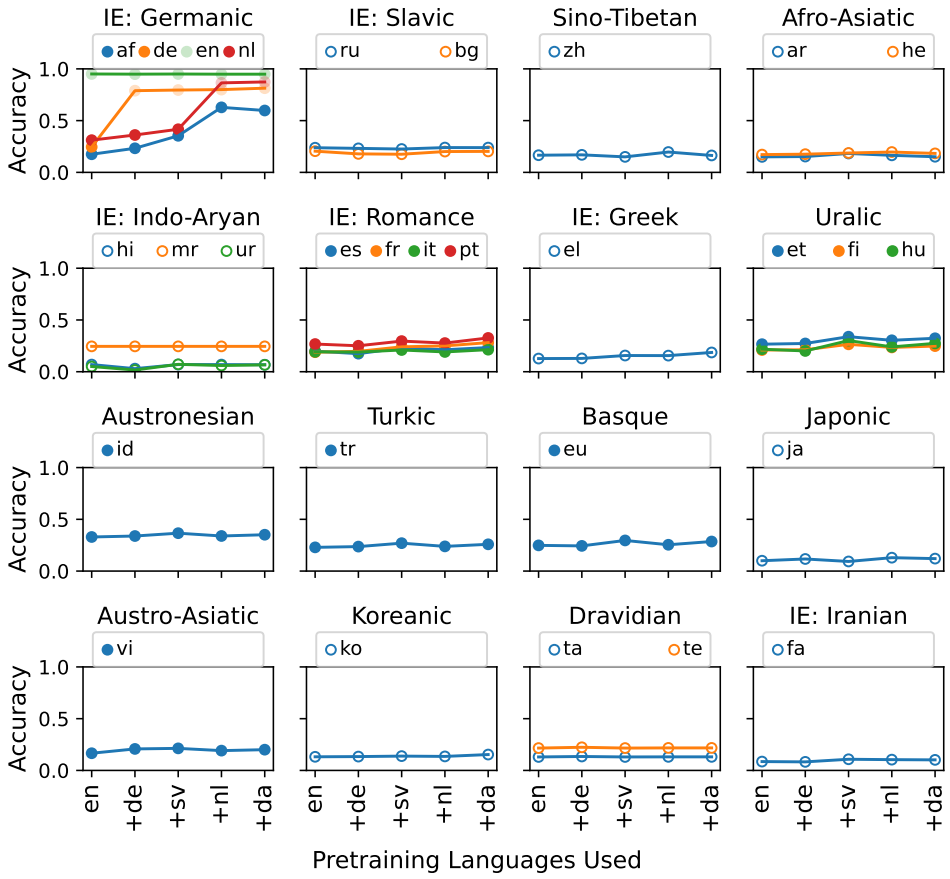


Figure 5.6: POS tagging accuracy using related pretraining languages (EN, DE, SV, NL, DA) grouped by families of target languages, with Indo-European (IE) languages further divided into subgroups following the XTREME dataset. The change in accuracy is mainly in Germanic, Romance, and Uralic languages due to only using pretraining languages from Germanic family.

Chapter 6

Conclusion

This dissertation describes methods to analyze and apply cross-lingual or multilingual models, focusing on cross-lingual word embeddings and multilingual language models. Chapter 3 sheds the light on evaluating cross-lingual word embeddings in a practical setting, i.e., without using evaluation data. The proposed resource-free metric based on graph modularity is experimented on multiple downstream tasks using cross-lingual word embeddings and shows that the graph modularity correlates well to multiple tasks.

Chapter 4 shows that cross-lingual word embeddings can be further improved by methods such as including graph modularity into the objective function (Section 4.1), using the graph encoder to retrofit (Section 4.2), and retrofit by simply overfitting to the training dictionary (Section 4.3). Also, further tuning the model towards a specific task, namely difficulty or readability estimation task, further improves cross-lingual transfer (Section 4.4).

Chapter 5 shows that cross-lingual transfer on languages unseen during pretraining are largely affected by the pretraining languages. Specifically, given restrictions on computational resources and experimenting on POS tagging, named entity recognition, and natural language inference, the findings are as follows: (1) choosing a diverse set of pretraining languages are preferred than the related pretraining languages, (2) without the model adaptation to the target languages, cross-lingual zero-shot scores in downstream tasks quickly plateaus after adding pretraining languages, and (3) the model adaptation to the target languages is crucial to exploit additional pretraining languages.

6.1 Future Directions

There are multiple potential future directions or topics in cross-lingual transfer or multilingual learning that are not covered in this dissertation.

Granularity of Tokens Models mainly discussed in this dissertation such as cross-lingual word embeddings, BERT, and RoBERTa use either word- or subword-level input tokens. Recent work by Rust et al. (2021) show that the choice of vocabulary for subword-level models is a key factor for training a better pretrained model for downstream tasks. On a different perspective, considering character- or byte-level inputs can help reduce the number of parameters of models (Tay et al., 2021b) but it can also potentially help generalize pretrained language models on languages unseen during pretraining. However, the downside of naively replacing units of tokens to characters or bytes is the longer input sequences. This makes it challenging for Transformers because the computation time increases by $O(N^2)$ for processing an input of N tokens, and further relating this topic to recent research on Transformers that can process longer input sequences with less computation time (Beltagy et al., 2020; Zaheer et al., 2020).

Beyond BERT and RoBERTa A more recent model, namely T5 (Raffel et al., 2020), has some notable differences when compared to BERT and RoBERTa. T5 is an encoder-decoder model while BERT and RoBERTa are known as "encoder-only" models (Raffel et al., 2020). The decoders generate texts in arbitrary length rather than predicting the probability of a single masked token, which makes encoder-decoder models a more suitable choice for generation tasks such as summarization and machine translation. Furthermore, T5 is trained on a slightly different pretraining task, which tries to predict a span (i.e., multiple consecutive tokens) from a single masked token[1] unlike MLM which predicts a single token from a single masked token. There are also publicly available T5 models which are jointly trained on multiple languages (Xue et al., 2021b). This opens up further research on the optimal choice of pretraining task in cross-lingual settings and analysis of pretrained multilingual encoder-decoder models on generation tasks.

[1] This is also known as "text infilling" (Lewis et al., 2020). Other options to corrupt a span in English are explored in depth by Lewis et al. (2020) and Raffel et al. (2020).

Creating Resources for Low-Resource Languages One more important future direction is the creation of resources in low-resource languages. Although one of the goals of research in zero-shot cross-lingual transfer is to build an NLP model without using labeled data from target languages, creation of datasets in multiple languages (e.g,. Ebrahimi et al. (2021)) is crucial to both evaluate and train better cross-lingual NLP models. For evaluation, it's possible to use resource-free evaluation methods (e.g., Chapter 3), however, these evaluation methods also have limitations and having real data is ideal. An interesting area in this topic is reducing annotator effort to create resources. This is important especially due to less number of speakers and can potentially have harder access to input devices. Few potential directions exist for reducing annotator efforts, ranging from active learning (Settles, 2009) to having an efficient annotation format (Sakaguchi et al., 2014).

Bibliography

Wasi Ahmad, Zhisong Zhang, Xuezhe Ma, Eduard Hovy, Kai-Wei Chang, and Nanyun Peng. 2019a. On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Wasi Uddin Ahmad, Zhisong Zhang, Xuezhe Ma, Kai-Wei Chang, and Nanyun Peng. 2019b. Cross-lingual dependency parsing with unlabeled auxiliary languages. In Conference on Computational Natural Language Learning.

Hanan Aldarmaki and Mona Diab. 2019. Context-aware cross-lingual mapping. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

David Alfter and Elena Volodina. 2018. Towards single word lexical complexity prediction. In Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications.

David Alvarez-Melis and Tommi Jaakkola. 2018. Gromov-wasserstein alignment of word embedding spaces. In Proceedings of Empirical Methods in Natural Language Processing.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A Smith. 2016. Massively multilingual word embeddings. Computing Research Repository, arXiv:1602.01925. Version 2.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016. A latent variable model approach to pmi-based word embeddings. Transactions of the Association for Computational Linguistics, 4:385–399.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2016. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In Proceedings of Empirical Methods in Natural Language Processing.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2017. Learning bilingual word embeddings with (almost) no bilingual data. In Proceedings of the Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In Proceedings of the Association for Computational Linguistics.

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In Proceedings of the Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. arXiv:2004.05150.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. Journal of Machine Learning Research, 3:1137–1155.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. Journal of Machine Learning Research, 3:993–1022.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 5.

Annette Capel. 2010. A1–B2 vocabulary: insights and issues arising from the English Profile Wordlists project. English Profile Journal, 1.

Annette Capel. 2012. Completing the english vocabulary profile: C1 and C2 vocabulary. English Profile Journal, 3.

Ethan C. Chau, Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank.

Xilun Chen and Claire Cardie. 2018. Unsupervised multilingual word embeddings. In Proceedings of Empirical Methods in Natural Language Processing.

Usman W Chohan. 2021. The challenge of selecting hyperpolyglots: A HYPIA approach. The Papers of the International Association of Hyperpolyglots (HYPIA).

Christos Christodouloupoulos and Mark Steedman. 2015. A massively parallel corpus: The Bible in 100 languages. Language Resources and Evaluation, 49(2).

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. 2017. Parseval networks: Improving robustness to adversarial examples. In Proceedings of the International Conference of Machine Learning.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2021. CANINE: Pre-training an efficient tokenization-free encoder for language representation.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In Proceedings of Empirical Methods in Natural Language Processing.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the International Conference of Machine Learning.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In Proceedings of the Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018a. Word translation without parallel data. In Proceedings of the International Conference on Learning Representations.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018b. XNLI: Evaluating cross-lingual sentence representations. In Proceedings of Empirical Methods in Natural Language Processing.

Council of Europe. 2001. Common European Framework of Reference for Languages: Learning, Teaching, Assessment. Press Syndicate of the University of Cambridge.

Brent Culligan. 2015. A comparison of three test formats to assess word difficulty. Language Testing, 32(4):503–520.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. Journal Of The American Society For Information Science, 41(6):391–407.

Tovly Deutsch, Masoud Jasbi, and Stuart Shieber. 2020. Linguistic features for readability assessment. In Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications.

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 North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. 2014. Improving zero-shot learning by mitigating the hubness problem. In Proceedings of the International Conference on Learning Representations.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In Proceedings of the International Conference on Learning Representations.

Matthew S. Dryer. 2013. Order of adposition and noun phrase. In Matthew S. Dryer and Martin Haspelmath, editors, The World Atlas of Language Structures Online. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Philipp Dufter and Hinrich Schütze. 2020. Identifying elements essential for BERT's multilinguality. In Proceedings of Empirical Methods in Natural Language Processing.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig. 2021. Ethnologue: Languages of the World, 24 edition. SIL International, Dallas.

Abteen Ebrahimi and Katharina Kann. 2021. How to adapt your pretrained multilingual model to 1600 languages. In Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing.

Abteen Ebrahimi, Manuel Mager, Arturo Oncevay, Vishrav Chaudhary, Luis Chiruzzo, Angela Fan, John Ortega, Ricardo Ramos, Annette Rios, Ivan Vladimir, Gustavo A. Giménez-Lugo, Elisabeth Mager, Graham Neubig, Alexis Palmer, Rolando A. Coto Solano, Ngoc Thang Vu, and Katharina Kann. 2021. AmericasNLI: Evaluating zero-shot natural language understanding of pretrained multilingual models in truly low-resource languages.

Julian Eisenschlos, Sebastian Ruder, Piotr Czapla, Marcin Kadras, Sylvain Gugger, and Jeremy Howard. 2019. MultiFiT: Efficient multi-lingual language model fine-tuning. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In Conference of the North American Chapter of the Association for Computational Linguistics.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In Proceedings of the European Chapter of the Association for Computational Linguistics.

Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. 2010. A comparison of features for automatic readability assessment. In Proceedings of International Conference on Computational Linguistics.

John Rupert Firth. 1957. A synopsis of linguistic theory 1930-55. 1952-59:1–32.

Thomas François, Nùria Gala, Patrick Watrin, and Cédrick Fairon. 2014. FLELex: a graded lexical resource for French foreign learners. In Proceedings of the Language Resources and Evaluation Conference.

Thomas François, Elena Volodina, Ildikó Pilán, and Anaïs Tack. 2016. SVALex: a CEFR-graded lexical resource for Swedish foreign and second language learners. In Proceedings of the Language Resources and Evaluation Conference.

Yoshinari Fujinuma, Jordan Boyd-Graber, and Michael J. Paul. 2019. A resource-free evaluation metric for cross-lingual word embeddings based on graph modularity. In Proceedings of the Association for Computational Linguistics.

Yoshinari Fujinuma and Alvin Grissom II. 2017. Substring frequency features for segmentation of Japanese katakana words with unlabeled corpora. In International Joint Conference on Natural Language Processing.

Yoshinari Fujinuma and Masato Hagiwara. 2021. Semi-supervised joint estimation of word and document readability. In Proceedings of the Fifteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-15).

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In Proceedings of the International Conference of Machine Learning.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A deep semantic natural language processing platform.

Filip Ginter, Jan Hajič, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 shared task - automatically annotated raw texts and word embeddings. LINDAT/CLARIAH-CZ digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Goran Glavas, Robert Litschko, Sebastian Ruder, and Ivan Vulic. 2019. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In Proceedings of the Association for Computational Linguistics.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In Proceedings of the Language Resources and Evaluation Conference.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In Proceedings of the Language Resources and Evaluation Conference.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In Proceedings of Empirical Methods in Natural Language Processing.

Sonal Gupta and Christopher D. Manning. 2015. Distributed representations of words to guide bootstrapped entity classifiers. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Masato Hagiwara and Satoshi Sekine. 2013. Accurate word segmentation using transliteration and language model projection. In Proceedings of the Association for Computational Linguistics.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In Proceedings of Empirical Methods in Natural Language Processing.

Harald Hammarström, Robert Forkel, Martin Haspelmath, and Sebastian Bank. 2021. Glottolog 4.4. Leipzig.

Shudong Hao. 2019. Crosslingual Topic Transfer. Ph.D. thesis, University of Colorado Boulder.

Zellig Harris. 1954. Distributional structure. Word, 10(23):146–162.

Michael Heilman, Le Zhao, Juan Pino, and Maxine Eskenazi. 2008. Retrieval of reading materials for vocabulary and reading practice. In Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications, pages 80–88.

Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. Neural computation, 18(7):1527–1554.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Computation, 9(8):1735–1780.

Yedid Hoshen and Lior Wolf. 2018. Non-adversarial unsupervised word translation. In Proceedings of Empirical Methods in Natural Language Processing.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In Proceedings of the International Conference of Machine Learning.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In Proceedings of Empirical Methods in Natural Language Processing.

Dan Jurafsky and James H. Martin. 2020. Speech and Language Processing (3rd ed. draft), 3 edition.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. 2020. Cross-lingual ability of multilingual BERT: An empirical study. In Proceedings of the International Conference on Learning Representations.

Nobuhiro Kaji and Masaru Kitsuregawa. 2014. Accurate word segmentation and POS tagging for Japanese microblogs: Corpus annotation and joint modeling with lexical normalization. In Proceedings of Empirical Methods in Natural Language Processing.

Thomas N. Kipf and Max Welling. 2016. Variational graph auto-encoders. NIPS Workshop on Bayesian Deep Learning.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In Proceedings of the International Conference on Learning Representations.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. 2012. Inducing crosslingual distributed representations of words. In Proceedings of International Conference on Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Proceedings of Empirical Methods in Natural Language Processing.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In Proceedings of Empirical Methods in Natural Language Processing.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In Proceedings of Advances in Neural Information Processing Systems.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In Proceedings of the International Conference on Learning Representations.

Anne Lauscher, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In Proceedings of Empirical Methods in Natural Language Processing.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In Proceedings of Advances in Neural Information Processing Systems.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. 2004. RCV1: A new benchmark collection for text categorization research. Journal of Machine Learning Research, 5.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the Association for Computational Linguistics.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2020. On the language neutrality of pre-trained multilingual representations.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In Proceedings of the Association for Computational Linguistics.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In Proceedings of the International Conference on Learning Representations.

Patrick Littell, David R Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In Proceedings of the European Chapter of the Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. RoBERTa: A robustly optimized BERT pretraining approach. In Proceedings of the International Conference on Learning Representations.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics.

Frederic M. Lord. 1980. Applications of Item Response Theory To Practical Testing Problems. Lawrence Erlbaum Associates.

Ilya Loshchilov and Frank Hutter. 2019. Fixing weight decay regularization in adam. In Proceedings of the International Conference on Learning Representations.

Matej Martinc, Senja Pollak, and Marko Robnik-Sikonja. 2019. Supervised and unsupervised neural approaches to text readability. CoRR, abs/1907.11779.

Héctor Martínez Alonso, Anders Johannsen, Sussi Olsen, Sanni Nimb, Nicolai Hartvig Sørensen, Anna Braasch, Anders Søgaard, and Bolette Sandford Pedersen. 2015. Supersense tagging for Danish. In Proceedings of the Nordic Conference of Computational Linguistics.

Héctor Martınez Alonso, Anders Johannsen, Sussi Olsen, Sanni Nimb, and Bolette Sandford Pedersen. 2016. An empirically grounded expansion of the supersense inventory. In Proceedings of the Global Wordnet Conference.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In Proceedings of Advances in Neural Information Processing Systems.

Arya D. McCarthy, Rachel Wicks, Dylan Lewis, Aaron Mueller, Winston Wu, Oliver Adams, Garrett Nicolai, Matt Post, and David Yarowsky. 2020. The Johns Hopkins University Bible corpus: 1600+ tongues for typological exploration. In Proceedings of the Language Resources and Evaluation Conference.

Vincent Micheli, Martin d'Hoffschmidt, and François Fleuret. 2020. On the importance of pre-training data volume for compact language models. In Proceedings of Empirical Methods in Natural Language Processing.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. Computing Research Repository, arXiv:1309.4168. Version 1.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In Proceedings of Advances in Neural Information Processing Systems.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In Proceedings of the Human Language Technology Conference.

David Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In Proceedings of Empirical Methods in Natural Language Processing.

Simonetta Montemagni, Francesco Barsotti, Marco Battista, Nicoletta Calzolari, Ornella Corazzari, Alessandro Lenci, Antonio Zampolli, Francesca Fanciulli, Maria Massetani, Remo Raffaelli, Roberto Basili, Maria Teresa Pazienza, Dario Saracino, Fabio Zanzotto, Nadia Mana, Fabio Pianesi, and Rodolfo Delmonte. 2003. Building the Italian syntactic-semantic treebank. In Treebanks: Building and Using Parsed Corpora. Springer.

Hajime Morita, Daisuke Kawahara, and Sadao Kurohashi. 2015. Morphological analysis for unsegmented languages using recurrent neural network language model. In Proceedings of Empirical Methods in Natural Language Processing.

Benjamin Müller, Antonis Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2020. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. CoRR, abs/2010.12858.

Sebastian Nagel. 2016. CC-News. `http://commoncrawl.org/2016/10/newsdataset-available`.

Ndapa Nakashole. 2018. NORMA: Neighborhood sensitive maps for multilingual word embeddings. In Proceedings of Empirical Methods in Natural Language Processing.

Ndapa Nakashole and Raphael Flauger. 2018. Characterizing departures from linearity in word translation. In Proceedings of the Association for Computational Linguistics.

Masashi Negishi, Tomoko Takada, and Yukio Tono. 2013. A progress report on the development of the CEFR-J. In Exploring language frameworks: Proceedings of the ALTE Kraków Conference, pages 135–163.

Graham Neubig. 2021. CS11-747 Neural Networks for NLP Multilingual Learning. `http://www.phontron.com/class/nn4nlp2021/assets/slides/nn4nlp-21-multilingual.pdf`. Accessed: 2020–11-06.

Mark E. J. Newman. 2003. Mixing patterns in networks. Physical Review E, 67(2).

Mark E. J. Newman. 2004. Analysis of weighted networks. Physical Review E, 70(5).

Mark E. J. Newman. 2006a. Finding community structure in networks using the eigenvectors of matrices. Physical Review E, 74(3).

Mark E. J. Newman. 2006b. Modularity and community structure in networks. Proceedings of the National Academy of Sciences, 103(23).

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. 2019a. Universal dependencies 2.4. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Joakim Nivre, Mitchell Abrams, Željko Agić, and et al. 2019b. Universal dependencies 2.5. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In Proceedings of Empirical Methods in Natural Language Processing.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In Conference of the North American Chapter of the Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In Proceedings of Empirical Methods in Natural Language Processing.

Telmo Pires, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In Proceedings of the Association for Computational Linguistics.

Tobias Plötz and Stefan Roth. 2018. Neural nearest neighbors networks. In Advances in Neural Information Processing Systems (NeurIPS).

Yada Pruksachatkun, Jason Phang, Haokun Liu, Phu Mon Htut, Xiaoyi Zhang, Richard Yuanzhe Pang, Clara Vania, Katharina Kann, and Samuel R. Bowman. 2020. Intermediate-task transfer learning with pretrained language models: When and why does it work? In Proceedings of the Association for Computational Linguistics.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research, 11:2487–2531.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1–67.

Leanne Rolston and Katrin Kirchhoff. 2016. Collection of bilingual data for lexicon transfer learning. UWEE Technical Report.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2017. A survey of cross-lingual embedding models. Computing Research Repository, arXiv:1706.04902. Version 2.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. In Proceedings of the Ninth Workshop on Statistical Machine Translation.

Guillaume Salha, Romain Hennequin, and Michalis Vazirgiannis. 2019. Keep it simple: Graph autoencoders without graph convolutional networks. Workshop on Graph Representation Learning, 33rd Conference on Neural Information Processing Systems (NeurIPS).

Peter H. Schönemann. 1966. A generalized solution of the orthogonal procrustes problem. Psychometrika, 31(1).

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In International Conference on Acoustics, Speech and Signal Processing.

Tal Schuster, Ori Ram, Regina Barzilay, and Amir Globerson. 2019. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In Proceedings of the Association for Computational Linguistics.

Burr Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Burr Settles, Geoffrey T. LaFlair, and Masato Hagiwara. 2020. Machine learning–driven language assessment. Transactions of the Association for Computational Linguistics, 8:247–263.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In Proceedings of the International Conference on Learning Representations.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In Proceedings of the Association for Computational Linguistics.

Stephanie Strassel and Jennifer Tracey. 2016. LORELEI language packs: Data, tools, and resources for technology development in low resource languages. In Proceedings of the Language Resources and Evaluation Conference.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In Proceedings of the Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Jai Gupta, Dara Bahri, Vamsi Aribandi, Zhen Qin, and Donald Metzler. 2021a. Are pre-trained convolutions better than pre-trained transformers?

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2021b. Charformer: Fast character transformers via gradient-based subword tokenization.

Trieu H. Trinh and Quoc V. Le. 2019. A simple method for commonsense reasoning.

Yulia Tsvetkov, Manaal Faruqui, and Chris Dyer. 2016. Correlation-based intrinsic evaluation of word vector representations. In Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of word vector representations by subspace alignment. In Proceedings of Empirical Methods in Natural Language Processing.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In Proceedings of the Association for Computational Linguistics.

Sowmya Vajjala and Detmar Meurers. 2012. On improving the accuracy of readability classification using insights from second language acquisition. In Proceedings of the Seventh Workshop on Building Educational Applications Using NLP.

Sowmya Vajjala and Taraka Rama. 2018. Experiments with universal CEFR classification. In Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications.

Shikhar Vashishth, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya, and Partha Talukdar. 2019. Incorporating syntactic and semantic information in word embeddings using graph convolutional networks. In Proceedings of the Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of Advances in Neural Information Processing Systems.

Elena Volodina and Sofie Johansson Kokkinakis. 2012. Introducing the Swedish Kelly-list, a new lexical e-resource for Swedish. In Proceedings of the Language Resources and Evaluation Conference.

Shuhan Wang and Erik Andersen. 2016. Grammatical templates: Improving text difficulty evaluation for language learners. In Proceedings of International Conference on Computational Linguistics.

Zihan Wang, Karthikeyan K, Stephen Mayhew, and Dan Roth. 2020a. Extending multilingual BERT to low-resource languages.

Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020b. On negative interference in multilingual models: Findings and a meta-learning treatment. In Proceedings of Empirical Methods in Natural Language Processing.

Felix Wu, Tianyi Zhang, Amauri Holanda de Souza Jr., Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying graph convolutional networks. In Proceedings of the International Conference of Machine Learning.

Shijie Wu, Alexis Conneau, Haoran Li, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Emerging cross-lingual structure in pretrained language models. In Proceedings of the Association for Computational Linguistics.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of BERT. In Proceedings of the Conference on Empirical Methods in Natural Language Processing and the International Joint Conference on Natural Language Processing.

Shijie Wu and Mark Dredze. 2020. Are all languages created equal in multilingual BERT? In Proceedings of the 5th Workshop on Representation Learning for NLP.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Menglin Xia, Ekaterina Kochmar, and Ted Briscoe. 2016. Text readability assessment for second language learners. In Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Ruochen Xu, Yiming Yang, Naoki Otani, and Yuexin Wu. 2018. Unsupervised cross-lingual transfer of word embedding spaces. In Proceedings of Empirical Methods in Natural Language Processing.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021a. ByT5: Towards a token-free future with pre-trained byte-to-byte models.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021b. mt5: A massively multilingual pre-trained text-to-text transformer.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In Association for the Advancement of Artificial Intelligence.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo Paetzold, Lucia Specia, Sanja Štajner, Anaïs Tack, and Marcos Zampieri. 2018. A report on the complex word identification shared task 2018. In Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, and Amr Yang, Li a nd Ahmed. 2020. Big bird: Transformers for longer sequences. In Proceedings of Advances in Neural Information Processing Systems, volume 33. Curran Associates, Inc.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Adversarial training for unsupervised bilingual lexicon induction. In Proceedings of the Association for Computational Linguistics.

Mozhi Zhang, Yoshinari Fujinuma, Michael J. Paul, and Jordan Boyd-Graber. 2020. Why overfitting isn't always bad: Retrofitting cross-lingual word embeddings to dictionaries. In Proceedings of the Association for Computational Linguistics.

Mozhi Zhang, Keyulu Xu, Ken-ichi Kawarabayashi, Stefanie Jegelka, and Jordan Boyd-Graber. 2019. Are girls neko or shōjo? cross-lingual alignment of non-isomorphic embeddings with iterative normalization. In Proceedings of the Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books.