Bilingual Lexical Induction for English-Sinhala

Anushika Liyanage

208033U

Thesis/Dissertation submitted in partial fulfilment of the requirements for the degree Master of Science in Computer Science and Engineering

Department of Computer Science & Engineering

University of Moratuwa
Sri Lanka

October 2022

DECLARATION

I, Anushika Liyanage, declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: UOM Verified Signature Date: 07/10/2022

The above candidate has carried out research for the Master's thesis/Dissertation under my supervision.

Name of Supervisor: Dr. Surangika Ranathunga

Signature of the Supervisor: UOM Verified Signature Surangika Ranathunga Digitally signed by Surangika Ranathunga Date: 2023.02.04 12:40:33 +05'30' Date: 01/02/2023

Name of Supervisor: Dr. Sanath Jayasena

Signature of the Supervisor: UOM Verified Signature Date: 01/02/2023

ABSTRACT

Bilingual Lexicons are important resources appertaining to Natural Language Processing (NLP) applications such as Neural Machine Translation and Named Entity Recognition (NER). However, Low Resource Languages (LRLs) equivalent to Sinhala lack such resources. Manually producing millions of word translations between languages is exhaustive and almost impossible. An increasingly popular approach to automatically create such resources is Bilingual Lexical Induction (BLI).

We created the first-ever BLI model for English and Sinhala language pair using the existing popular model VecMap. Currently, no prior work has conducted a sufficient evaluation with respect to the factors, nature of the dataset, type of embedding model used, or the type of evaluation dictionary used on BLI and how these factors affect the results of BLI. We fill the gap by executing an extensive set of experiments with regard to the aforementioned factors on BLI for Sinhala and English in this thesis.

Furthermore, we enhance the pre-trained embeddings to cater to the application by applying sophisticated post-processing approaches. Linear transformation and effective dimensionality reduction are applied to the pre-trained embeddings before obtaining cross-lingual word embeddings between Sinhala and English by applying VecMap. Furthermore, we have introduced dimensionality reduction to the VecMap algorithm where the algorithm starts the first iteration from a low dimension to initialize a better solution. Subsequently, the dimensionality of the embeddings is increased in each iteration until embeddings reach the original dimension in the final iteration. We were able to improve the results considerably by learning a better initial solution and hence an improved final solution. Finally, we combined the post-processing step with the modified VecMap model to obtain even better mapping for Sinhala-English language pair which in turn is applicable in task-specific downstream systems to improve the results of the entire system.

Keywords: Sinhala; embedding spaces; embedding models; bilingual lexicon induction

ACKNOWLEDGEMENTS

I would like to recognize the invaluable guidance of my supervisors, Dr.Surangika Ranathunga and Prof.Sanath Jayasena. I have greatly benefited from their insights, vast knowledge, skillful supervision, and suggestions made throughout this research. I am very thankful for the constant support, patience, understanding, and motivation extended to me when times were difficult.
I would like to thank Prof. Gihan Dias, for the critical advice given throughout this research. I would like to extend my gratitude to the academic and non-academic staff of the Department of Computer Science and Engineering for providing the required resources to carry out this research. This research was funded by the University of Moratuwa AHEAD project research grant.
I thank my partner Milinda Rajapakshe for listening to my constant rant, unconditional love, support, and for the sacrifices you made in order for me to pursue a master's degree. I am forever grateful to my friends and family for the love, support, and motivation in this entire thesis process every day.

Thank you!

LIST OF ABBREVIATIONS

NLP	Natural Language Processing
NMT	Neural Machine Translation
NER	Named Entity Recognition
BLI	Bilingual Lexical Induction
PCA	Principal Component Analysis
PPA	Post Processing Algorithm
RNN	Recurrent Neural Networks
SA	Sentiment Analysis
LRL	Low Resource Languages
HRL	High Resource Languages
SVD	Singular Value Decomposition
NN	Nearest Neighbor

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Background

A bilingual dictionary is a glossary that consists of a list of words and phrases in one language, along with respective translations in another language. A similar resource is a bilingual lexicon, which can be identified as a specific vocabulary or a list of word translations between two languages in a particular field or a domain such as medicine, engineering, psychology, administrative etc [3, 4, 5]. Bilingual lexicons and dictionaries were predominantly generated manually by experts in both languages. Existing bilingual dictionaries/ bilingual lexicons consist of a limited number of word translation pairs.
Bilingual dictionaries/lexicons are considered an important resource in NLP since such resources can be used for numerous NLP applications similar to NMT and NER [6, 7, 8]. However, languages consist of millions of words and manually translating all the words is infeasible and almost impossible. Additionally, the translation/meaning of a word can differ from the context in which the word is used.
ex: word May

- May he rests in peace
- It will commence on the 01st of May 2020

Therefore even the translations done by experts can be considered erroneous when applied in different domains or contexts. This will make it incorrect to use such dictionaries in domain-specific downstream systems and indicates the importance of applying domain-specific lexicons for similar tasks. In addition, low-resource languages equivalent to Sinhala lack important resources such as bilingual dictionaries and lexicons and such resources are considered crucial for various NLP applications.

1.2 Research Problem

An increasingly popular approach to alleviating the above issues is Bilingual Lexical Induction. BLI is the task of automatically inducing word translations between two languages using a monolingual corpus [9]. There are many different techniques introduced and implemented for BLI. However, none of them was adapted or implemented for the Sinhala-English language pair. LRLs are languages that lack some aspects such as lack of a unique writing system or stable orthography, lack of electronic resources for speech and language processing such as monolingual corpora, bilingual electronic dictionaries, and vocabulary lists [10]. Sinhala being a low-resource language, it is crucial for the Sinhala language to adapt such mechanisms to generate similar resources. Many existing approaches for BLI benefit from parallel corpora and supervision to obtain high-quality results. Even though such resources exist for High-Resource Languages(HRLs) it is challenging to produce similar resources for LRLs, making it difficult or outright impossible to apply similar approaches for LRLs. Hence it is critical to adapt an appropriate technique befitting the existing resources for LRLs similar to Sinhala. However, even LRLs can acquire comparable corpora between language pairs due to the abundance of resources available on the world wide web.

There are multiple solutions proposed to resolve the issues that arise when implementing BLI for LRLs due to a lack of resources. One such implementation is the well-known robust VecMap algorithm to find a cross-lingual mapping between embedding spaces to generate the bilingual lexical [11]. Since this method requires only monolingual data between two language pairs with no supervision VecMap algorithm can be considered a great starting point for LRL pairs.
Results of the BLI techniques can vary depending on the dataset used, the embedding model, or the evaluation dictionary. Therefore it is vital to be aware of the factors that affect the results of BLI and how those factors affect the results of BLI. Nevertheless, no comprehensive analysis has been carried out on the factors that can affect the results of BLI thus far. Comprehensive analysis of factors that

affect BLI has to be achieved by running an extensive set of experiments with multiple datasets, evaluation dictionaries and embedding models.
Moreover, how the BLI techniques are impacted by language-specific features such as inflections have to be studied. Finally, it is always possible to enhance the already existing BLI techniques for better performance.

1.3 Research Objectives

The objectives of this research are two-fold and as follows:

1. Adapt Bilingual Lexical Induction for English-Sinhala language pair
   (a) Adapt existing best suitable BLI model for Sinhala-English
   (b) conduct a comprehensive analysis of the factors that affect BLI
2. Further enhance the adapted model to cater Sinhala-English Language pair

1.4 Contributions

We make the following contributions in this thesis:

- Introducing the first-ever implementation of BLI for Sinhala-English language pair
- Conducted the first-ever comprehensive analysis of the factors that affect BLI
- Introduced different enhancements methods to pretrained word embeddings
- Introduced an enhancement to the existing mapping algorithm
- Introduced a novel approach for BLI by combining multiple enhancements on pretrained word embeddings and mapping algorithm

1.5 Publications

- **Anushika Liyanage**, Surangika Ranathunga, and Sanath Jayasena."Bilingual lexical induction for Sinhala-English using cross lingual embedding spaces." 2021 Moratuwa Engineering Research Conference (MERCon). IEEE, 2021. ***(published)***

Chapter 2

LITERATURE SURVEY

2.1 Overview

Several embedding models, algorithms, and enhancement techniques have been used in this study. They are discussed in this section. We start by discussing the vector representation of words. Then we discuss different Embedding models and their architecture. Next, we move on to explain the cross-lingual alignment of word vectors which is the technique used in this research to map embedding spaces. Subsequently, we discuss the VecMap model we adapted for the Sinhala-English language pair. We discuss how the VecMap model is the best suitable model to be adapted for Sinhala-English languages by comparison with a few other existing novel techniques. Afterwards, post-processing approaches on pre-trained embeddings, linear transformation, and effective dimensionality reduction using principal component analysis that can be applied to enhance the BLI results are discussed. Next, we conclude this section by discussing the main methods of generating BLI models along with different methodologies implemented to improve BLI, particularly the post-processing of embedding spaces and diverse strategies of post-processing. Furthermore, we discuss approaches such as improving word vectors to incorporate language-specific details similar to morphology. We will be discussing the different procedures and enhancements used for BLI to identify the feasibility of adapting the best suitable approaches for low-resource languages such as Sinhala. The models discussed in this section are referred to throughout the study.

2.2 Vector representation of words

Word vectors can be identified as an attempt to mathematically represent the meaning of a word. Word vectors are represented by multidimensional continuous floating-point numbers. Furthermore, semantically similar numbers are mapped close together in the geometric vector space [12]. Put differently, word vectors are real-valued numbers where each point in the vector space captures the dimension of the meaning of the words and semantically similar words are mapped together in the space. As can be observed in Figure 2.1 relatedness between words can be represented by the geometric relations of vectors. As an example, the cosine similarity between the words “Cat” and “Animal” is greater than the cosine similarity between the words “Cat” and “Car”.

The vector values (numbers) in a word vector can be identified as a word’s distributed weight across each dimension. Each dimension can be considered as a possible semantic meaning for the word. The number associated with each dimension captures the closeness of the word’s meaning to that dimension. Hence the semantics of the words are embedded across each dimension. There are two predominant methods to generate word vectors.

1. **Count based methods:** This method creates a co-occurrence matrix between each word and its context. This approach considers the words within a linear window in a manner, that the resultant matrix has a square shape. In this method dimension of the matrix will be equivalent to the number of words in the corpus. The counts and dimensions are then normalized, weighted and the context dimensionality of the space is reduced by different dimensionality reduction techniques similar to Principal Component Analysis(PCA) or its generalized version Singular Value Decomposition(SVD). Here PCA is used for the study of Principal Word Vectors [13] while SVD is used in the Latent Semantic Analysis model (LSA) [14]

2. **Prediction based methods:** This method initializes vectors randomly in a shallow neural network. Then the model trains the parameters to predict either the context, given the word, or the word, given the context. A few of the most prominent models of this method are the Continous Bag Of Words(CBOW) and Skip-Gram with Negative Sampling(SGNS)

models of word2vec embedding model [15] and Glove, which rely on global co-occurrence counts from the corpus for word representations [16].

In all the discussed methods, word embedding output represents the word with the corresponding vector in a matrix of vectors(Figure 2.3)

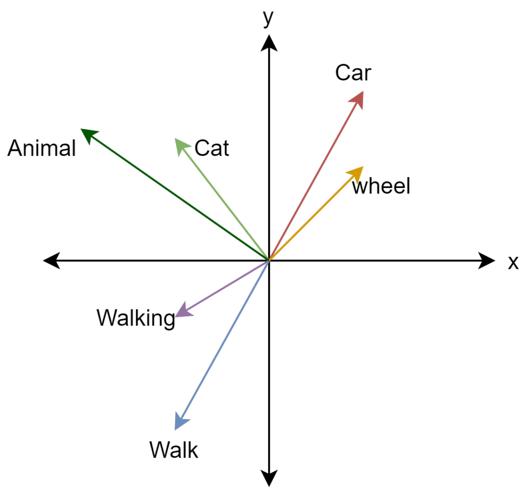
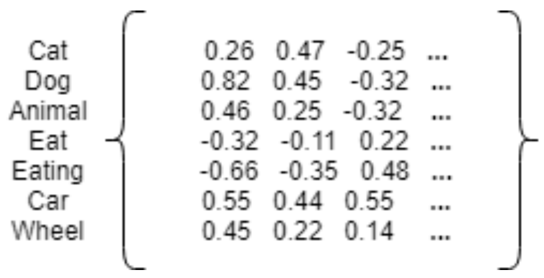


Figure 2.1: Illustrative example of vector representation of words in a two-dimensional space



$$\begin{array}{c} \text{Cat} \\ \text{Dog} \\ \text{Animal} \\ \text{Eat} \\ \text{Eating} \\ \text{Car} \\ \text{Wheel} \end{array} \left\{ \begin{array}{cccc} 0.26 & 0.47 & -0.25 & \ldots \\ 0.82 & 0.45 & -0.32 & \ldots \\ 0.46 & 0.25 & -0.32 & \ldots \\ -0.32 & -0.11 & 0.22 & \ldots \\ -0.66 & -0.35 & 0.48 & \ldots \\ 0.55 & 0.44 & 0.55 & \ldots \\ 0.45 & 0.22 & 0.14 & \ldots \end{array} \right\}$$

Figure 2.2: Simple word vector representation matrix where each row of the matrix represents a word vector and columns represent the dimensions

2.3 Embedding Models

In this study, we have used both Word2Vec and fastText embedding models to build our embeddings. Both these models are prediction-based.

2.3.1 Word2Vec Model

The word2vec model introduced by Mikolov et al. [15] brings forth state-of-the-art embeddings carrying a semantic representation of words. word2vec model consists of a shallow neural network model which is trained to reconstruct linguistic representations of words. Two different learning approaches were introduced as a part of the word2vec model.

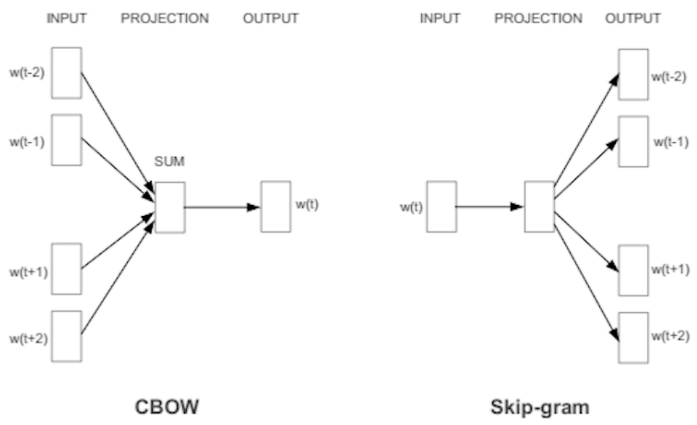


Figure 2.3: Architecture of the word2vec models: CBOW and Skip-Gram. Image source [1]

1. **Continuous Bag of words model (CBOW model)**

   This approach is similar to a feed-forward neural net language model architecture. The model attempt to predict the target when provided with a list of context words. The model basically considers the distribution of context words and predicts the target word.

2. **Skip-Gram model**

   This model acts as the inverse of the CBOW model. Given a word Skip-Gram model will try to predict the context words/surrounding words of the given word.

Hierarchical soft-max and negative sampling were later introduced as improvements to word2vec model [1].

2.3.2 FastText Model

fastText model is based on the skip-gram model and, yet unlike distinct vector representation for each word, the model assigns a vector for each character level n-grams. Each word is represented with a bag of character-level n-grams. Since each n-gram is represented by a vector, an entire word is represented by a sum of the mentioned character-level n-gram vectors [17]. This model allows to capture morphology and sub-wording information, thus can be expected to capture the similarity between orthographically similar words. As an example, the model will be able to capture the similarity between words such as "wash" and "washing" hence will be favourable for languages with a vast vocabulary and rare words.

2.4 Cross-lingual Alignment of Word Vectors

All common languages share concepts that are grounded in the real world. As an example, both cats and dogs are animals. Figure 2.4 is a visualization of vector representation for numbers and animals for English and Spanish languages reduced to two dimensions by PCA and manually rotated to accentuate the similarity. It can be easily observable that there is a geometric similarity between these two languages as mentioned above. Similarly, often there are strong geometric relationships between vector spaces of different languages [2].

As a consequence, it is possible to discover an accurate mapping between two languages to find translations by applying linear projection between vector spaces. More concretely a vector space X can be sent through a transformation matrix W such that it maps to a vector space Z where the spatial distance between X and Zs translation word pairs is minimum.

A different approach to linear projection approaches is to align cross-lingual word embeddings utilizing the iterative Procrustes analysis to find an optimal orthogonal mapping between languages. This approach relies on finding an orthogonal mapping between source space and target space such that the orthogonal distance between points will be minimum [18]. An orthogonal transformation is a linear transformation where orthonormal bases are mapped to orthonormal

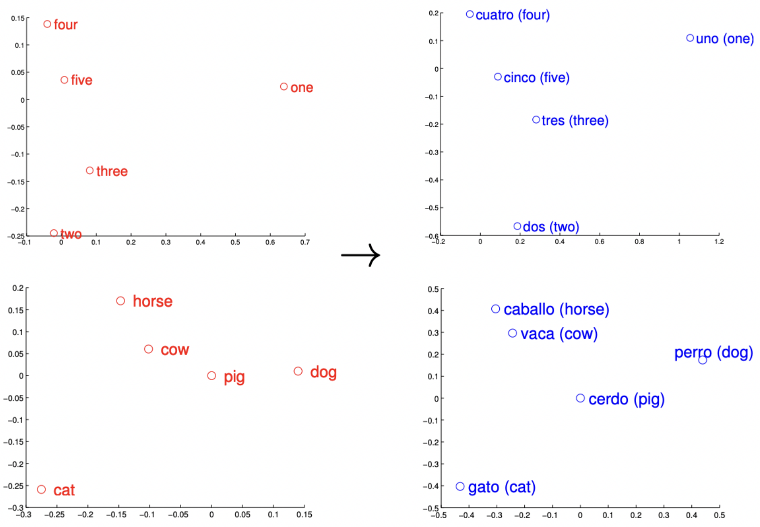


Figure 2.4: Geometric Similarity between English and Spanish. Image source [2]

bases. Orthogonal transformation preserves the length and angle between vectors. Matrix representation of an orthogonal transformation is an orthogonal matrix. Singular Value Decomposition is used to find the optimal orthogonal transformation as proven by Schonemann et al. [19]. When U and V^T are orthonormal unitary matrices and S is the diagonal matrix of singular values, SVD decomposition of matrix X:

$$X = USV^T \tag{2.1}$$

U and V^T approximately represent the directions of vertical and horizontal vectors respectively, which can be seen as rotations or reflection matrices. S determines the scaling of U and V^T and is necessary to reconstruct X.

SVD is used to find optimal orthogonal transformation to solve the orthogonal Procrustes problem. If X and Z are source and target matrices respectively with the same dimensionality and equal size, the orthogonal Procrustes problem can be used to find an orthogonal transformation. Schonemann et al. [19] proved that if R is transformation learned by solving the orthogonal Procrustes problem, RX' will be approximately equal to Z.

To find a rotation, almost all the existing methods for alignment methods require a bilingual lexicon with few term alignments [18, 15]. However, recent work has been able to minimize the size of the seed dictionary or eradicate the need for the dictionary completely [11, 20].

2.5 Post-Processing Embedding Spaces

Artetxe et al. [21] showed that post-processing of embedding spaces could lead to finding a better mapping between cross-lingual embedding spaces and which will result in a substantial improvement on BLI [22]. In this study, we incorporate two different approaches of post-processing to improve the results of BLI.

- **Generalization of first-order and second-order monolingual similarities to the nth-order similarity:** Artetxe et al. [21] argue that different word embedding models capture different divergent and often incompatible qualities such as semantic/syntactic information or similarity/relatedness measures. Hence different applications of word embeddings will require different models based on the information captured by the model.
  If X is the embedding matrix and X_{i*} is the ith word embedding in the matrix, then we can measure the similarity to some extent between the ith and jth word embeddings by taking the dot product between as $sim(i,j) = X_{i*}.X_{j*}$. Therefore we can define the first-order similarity matrix as $M(X) = XX^T$ where $sim(i,j) = M(X)_{i,j}$. Similarly, we can define second-order similarity on top of this first-order similarity. Here second-order similarity does not capture the similarity between the words i and j. Instead, it would capture the similarity of i and j to a third-word k, which in turn makes them comparably more similar in second-order similarity. Furthermore, one could likewise calculate the third, fourth similarity up to nth order similarity. The underlying idea is that some of these similarity orders will be more capable of capturing different aspects of language as mentioned.

- **Effective dimensionality reduction:** Word embedding models can be further improved by adding dimensionality reduction as a post-processing step [23]. We have used PCA one of the most common dimensionality reduction method along with a post-processing algorithm, in this research for dimensionality reduction. PCA is the orthogonal projection of the data into a lower-dimensional linear space and can be implemented using SVD [24].

2.6 Bilingual Lexical Induction

BLI can be supervised, semi-supervised, or unsupervised and currently there are three main approaches to generating BLI models. In this section, we discuss these three previous approaches and the different improvements proposed for those basic methods. Early techniques to generate BLI models depended heavily on count-based vector space models where translations are typically learned using parallel corpora and used various distributional similarity metrics/properties in the corpora as signals of equivalence. However, these approaches were replaced by rather sophisticated methods that induce joint cross-lingual embeddings to generate the lexical. These strategies continued to assume some level of supervision and solely relied on translating frequent words that appeared in the dataset. Later on, these approaches were superseded by projection-based or mapping approaches where the need for supervision is minimized or completely eradicated.

2.6.1 Count-based Vector Space Models

Count-based vector space models typically utilize two monolingual corpora from each language which can be completely unrelated or parallel along with a small seed dictionary for low supervision. Past work experimented with and relied on a variety of monolingual distributional similarity metrics as signals of equivalence between two languages including orthographic similarity, contextual similarity, temporal similarity etc [25, 26, 27]. Some of the prior work has combined different types of these orthogonal signals employing methods such as rank combination to obtain better results [28]. Based on the selected similarity metric, a vector space model is created using the monolingual corpora. For instance, if the contextual similarity is chosen as the signal of translation equivalence, then this method consists of finding a mapping between the context vector space of one language to another language. Once the context vectors are generated for each word in each language and projecting the vector space of the target language into the source language similarity between two words can be calculated using cosine similarity [27]. If the context vectors of i and j are W_i and W_j then the similarity between source word i and target word j $Sim(W_i, W_j)$ can be calculated from the below expression.

$$Sim(W_i, W_j) = (W_i.W_j)/||W_i||.||W_j|| \tag{2.2}$$

Similarly, if the temporal similarity is selected, then the usage of words over time is monitored as the signal of translation equivalence [29]. This is under the assumption that over time all over the world people tend to discuss similar topics in the same time frame in different languages hence will be able to find a mapping between monolingual data based on that. Furthermore, if the orthographic similarity is based on the assumption that similar words are spelt similarly in different languages, especially names and places. Hence by calculating the edit distance between two words by normalizing it by the lengths of the words will be able to obtain a measurement of orthographic similarity. Additionally, there are several different approaches used in count-based vector space models such as topic similarity, frequency similarity, burstiness similarity, etc. Above discussed methods require high levels of supervision firstly to identify beneficial signals of translations and then to map monolingual vector spaces of the selected languages. Irvin et al. [9] performed a comprehensive analysis of the above-discussed approaches and concluded that the accuracy of BLI is high for frequent words and terms with bursty nature. Furthermore, they reasoned that the results of BLI can be improved by increasing the monolingual data abundantly and by identifying the best-suited signals of translation equivalence.

2.6.2 Inducing Joint Cross-lingual Embedding Models

Subsequent work on BLI has focused on inducing joint cross-lingual embedding spaces utilizing bilingual corpora. Cross-lingual embedding models attempted to represent vocabularies of more than one language into one continuous common vector space.

These approaches heavily relied on different types of supervision from supervised to weakly supervised. Also, these approaches used different levels of supervision which are namely word level, sentence level, and document level.

There exist different approaches to induce a joint cross-lingual embedding space. Klementiev et al [30] used word co-occurrence statistics of parallel data as the signal of alignment and induce the joint cross-lingual embedding space. Gouws et al. [31] used distant supervision to build embeddings relying on small seed dictionaries, and they built a word embedding model such that similar words in different languages will be close to each other in the embedding space. This approach uses non-parallel corpora of two languages and will build task-specific embeddings, as an example for POS tagging, NER, or sentiment analysis. Early work of cross-lingual alignment methods were limited only to two languages rather than aligning multi-languages [32].

Experiments for Cross-lingual Embedding alignment approaches were largely conducted on HRLs due to the requirements of supervision and large bilingual datasets [33, 34, 35]. Hence only a handful of experiments were conducted on LRLs [36, 31]

2.6.3 Projection Based or Mapping Approaches

Above discussed approaches were superseded by recent methods of post-hoc alignment of word embeddings that were independently trained for each language. In projection-based approaches, the word embeddings for each language are trained independently utilizing monolingual corpora, and they were later aligned to find a mapping between the languages through a linear transformation. Projection-based approaches typically use a very small seed dictionary (starting from about

25-word translation pairs to a few thousand-word translation pairs) as the weak bilingual supervision. A vital component of this approach is the iterative refinement of the initial seed dictionary to obtain a better mapping/projection space between two languages [37, 1].

Most of the existing work for projection-based approaches directly retrieves nearest neighbours from the current shared vector space. Artetxe et al. [11] starts mapping with a weak initial solution and then iteratively improves the solution. Cosine similarity is taken as a measure of similarity and used to retrieve the nearest neighbours in their research. Artetxe et al. [11] have introduced fully unsupervised, weakly supervised, and supervised versions of the experiments. Later work on projection-based approaches was able to completely eradicate the need for bilingual supervision by using adversarial training or distribution-based approaches [38, 39]. Conneau et al. [38] introduced a fully unsupervised approach where they utilized adversarial training to find a linear mapping between two languages. The proposed method operates in two steps, where firstly mapped source embeddings and target embeddings are identified by a trained discriminator, and then using closed-Procrustes they improve the mapping by extracting a synthetic dictionary from the shared embedding space. This approach also relies on basic nearest neighbour extraction from the shared vector space.

In comparison, Karan et al. [40] introduced a novel classification step to each iteration of the self-learning process where before including a word pair into the training dictionary each translation pair will be going through a pretrained classifier. This will ultimately reduce the noised entries to the dictionary. Only a handful of work conducted experiments on LRLs and none has done an in-depth analysis on how the introduced methods will work on LRLs [38, 11].

2.6.4 VecMap Model

We are applying the recently introduced well-known VecMap model (Figure 2.5)in our research to find a cross-lingual alignment between Sinhala and English languages. The VecMap model trains monolingual embeddings for each language and then finds a mapping using a linear transformation. The algorithm suggests a fully unsupervised initialization to find the cross-lingual mappings and then, the algorithm iteratively improves the weak initial solution by combining it with a more robust self-learning method to obtain the optimum mapping between embedding spaces. VecMap model consists of four sequential steps namely pre-processing, initialization of the initial solution, iteratively improving the solution, and a final refinement of the solution [11].

- **Pre-processing:** This is the first step of the VecMap algorithm and in this step, both source and target embedding spaces are pre-processed to normalize the embeddings. Prepossessing starts by length normalizing the embeddings and mean centering each dimension. Then embeddings are length normalized again to ensure embeddings have a unit length. Having unit length for embeddings corroborates that the dot product between embeddings is equal to the cosine similarity and explicitly comparable to their Euclidean distance.

- **Initialization:** In this step algorithm learns a fully unsupervised initial solution. The challenging part of finding a mapping is that source embedding matrix X and target embedding matrix Z are not aligned across both axes, hence there's no direct correspondence between source and target languages. To overcome this issue and to build the initial solution algorithm, first construct alternative matrices if X and Y, which are aligned across their jth dimension as X' and Z' respectively. These embedding matrices are used to build the initial solution dictionary that aligns their respective vocabularies.

  If embedding spaces are perfectly isometric, similarity matrices of X $M_X = XX^T$ and similarity matrix of Z $M_Z = ZZ^T$ would be equivalent up to a permutation of their rows and columns. Based on the assumption that the isometry condition approximately holds in practice, one can try every possible permutation between M_X and M_Z to find the best possible map.

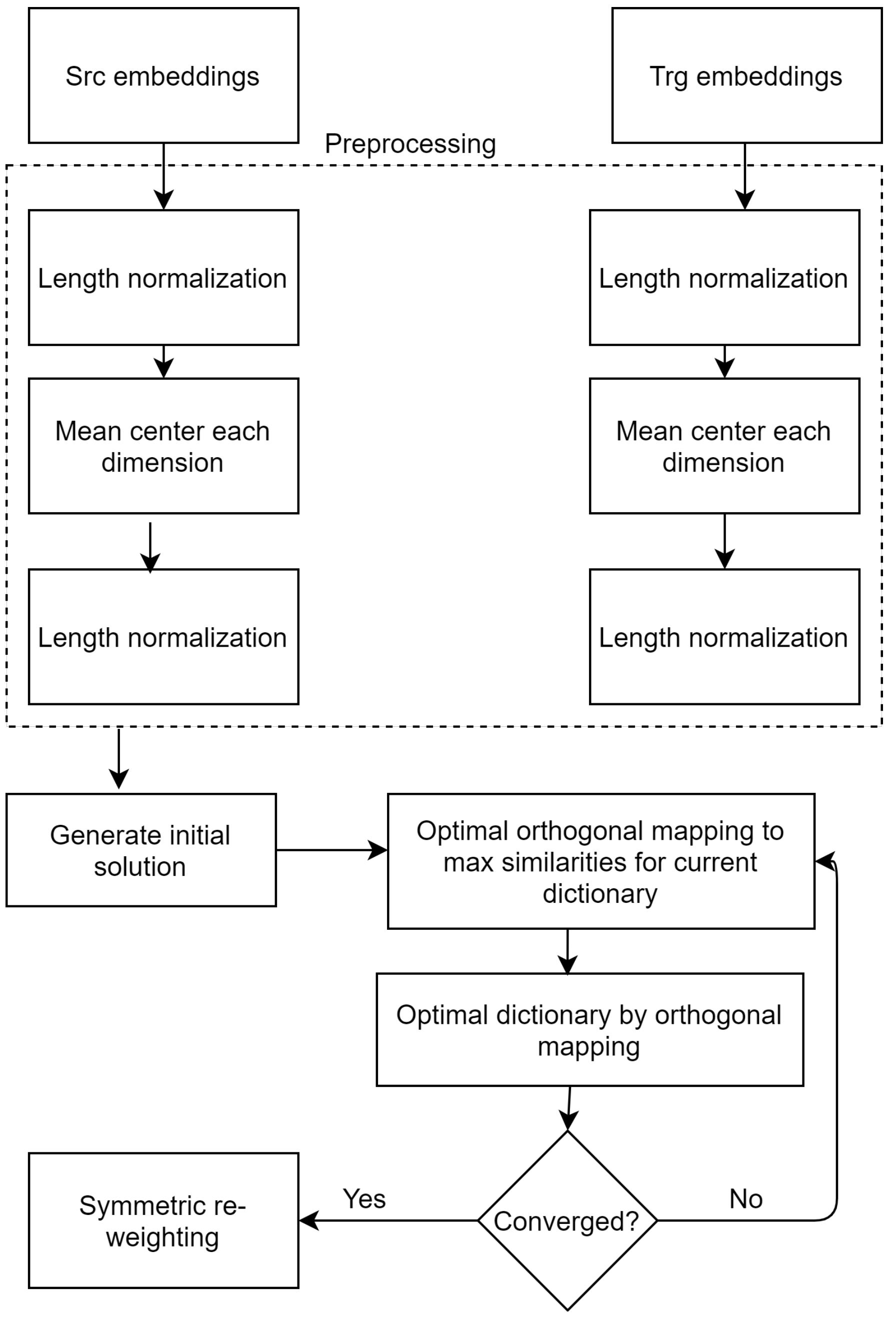

Src embeddings
Trg embeddings
Preprocessing
Length normalization
Length normalization
Mean center each dimension
Mean center each dimension
Length normalization
Length normalization
Generate initial solution
Optimal orthogonal mapping to max similarities for current dictionary
Optimal dictionary by orthogonal mapping
Symmetric re-weighting
Yes
Converged?
No

Figure 2.5: VecMap Model. Image source [3]

However, resulting combinations of permutations would make this approach intractable. As a mean to overcome this challenge, one could sort each row of M_X and M_Z matrices. For strictly isometric languages, equivalent words would have the exact same vector across languages, hence in an approximately isometric situation, we could find the corresponding translation of a word in sorted M_X by retrieving the nearest neighbour in sorted M_Z.

Furthermore if SVD of X is $X = USV^T$ then the similarity matrix $M_X = US^2U^T$. The square root of the Similarity matrix $M_X^{1/2} = USU^T$ is closer to original embeddings in nature and works better in practice. Hence to obtain the X' and Z' matrices to build the initial solution, the VecMap algorithm computes sorted $M_X^{1/2}$ and sorted $M_Z^{1/2}$ and normalizes the sorted embeddings as described in the first step.

- **Robust self-learning:** In this step algorithm iterate through the following steps until convergence.

  1. When X_{i*} is ith vocabulary item in X and Z_{j*} is jth vocabulary item in Z, Maximizing the similarities of current solution dictionary D by computing the optimal orthogonal mapping as follows.

     $$\underset{W_X,W_Z}{\text{argmax}} \sum_i \sum_j D_{ij}((X_{i*}W_X)(Z_{j*}W_z)) \tag{2.3}$$

     The optimal solution is given by $W_X = U$ and $W_Z = V$ when $USV^T = X^TDZ$ is the SVD of X^TDZ

  2. Utilizing the mapped embeddings $XW_XW_Z^TZ^T$ compute the optimal dictionary, which generally uses NN retrieval from source to target.

  In order to avoid algorithms stuck in poor local optima, the approach suggests a few key improvements namely stochastic dictionary induction, frequency-based vocabulary cutoffs, Cross-domain Similarity Local Scaling (CSLS) retrieval and bidirectional dictionary induction.

- **Symmetric re-weighting:** If SVD $USV^T = X^TDZ$ then $W_X = U$ and $W_Z = VS$ where X and Z are previously whitened by applying a linear transformation and later de-whitened [37]. Re-weighting is applied symmetrically to both languages by taking $W_X = US^{1/2}$ and $W_Z = VS^{1/2}$ after the self-learning converged to a good solution.

2.7 Post-Processing Embedding Spaces

Recently the application of pretrained embedding to numerous research fields such as topic modelling, parsing, named entity recognition, document classification and neural machine translation has largely grown [21]. This being the case, the importance of improving the quality of pretrained embeddings via different approaches became a novel research avenue. One of the increasingly popular approaches to enhance the quality of the pretrained embeddings is to post-process the pretrained embedding spaces. There has been a number of research with different techniques on post-processing pretrained monolingual embeddings. Faruqui et al. [41] try to enhance word embeddings by getting relational information from semantic lexicons and this converts embeddings into sparse representations which are more interpretable. Furthermore, antonymy and synonymy constraints are administered into vector space in the research conducted by Mrkvsic et al. [42] whereas Nguyen et al. [43] tries to reinforce weakening noise and salient information as a denoising step of the pretrained word embeddings.

Moreover, Labutov et al. [44] proposed a method to adapt existing embeddings to improve performance specifically in supervised tasks to achieve unconstrained optimization utilizing regularization. Rothen et al. [45] introduced a method, that learns orthographic transformations that will focus on the relevant task in a lower-dimensional subspace than the original vector space called ultra-dense subspace.

More recent work on post-processing pretrained embeddings by Artetxe et al. [21] argues that word embeddings consist of information that is not directly apparent and different embedding models capture different linguistic aspects such as similarity and relatedness aspects or semantic and syntactic aspects which are often mutually incompatible. They adjust the similarity order of the model using a linear transformation without utilizing any other external resources to achieve better results in different aspects depending on the application of embeddings. We can consider similarity/relatedness to be one axis and the semantic/syntax to be the other axis of the similarity Artetxe et al. [21] propose a methodology

to tailor any embedding space to lean on either of these axis's depending on the requirement of the application. In this research similarity order which is first or second-order co-occurrences were generalized and used as a continuous parameter for linear transformation applied to the embedding model.
Mu et al. [46] increase the discrimination of the embeddings by projecting the word vectors away from their most dominant direction to improve the performance considerably. Raunak et al. [23] introduces effective dimensionality reduction using PCA with the post-processing algorithm introduced by Mu et al. [46] to create a more efficient and effective set of embeddings with reduced dimension. They first create more discriminated embeddings by applying the post-processing algorithm proposed by Mu et al. [46] and then apply effective dimensionality reduction using PCA to the discriminated embeddings to obtain a set of low dimensional embeddings. Finally, they once again apply the same post-processing algorithm to the low dimensional embeddings to obtain a purified set of embeddings.

2.8 Summary

Although there has been various research conducted on different BLI techniques, only a handful of research has been conducted on low resource domain [38, 39, 47]. However, None of the existing research has conducted a deep comprehensive analysis of the contributing factors to BLI such as the type of the dataset, the evaluation dictionary used, or the type of the embedding model used. Furthermore, Vulic et al. [22] conducted improving BLI via combining post-processing technique linear transformation with cross-lingual embedding mapping there was no other research conducted combining multiple post-processing techniques with BLI or different post-processing techniques with BLI. There has been much research conducted on improving BLI using different techniques [37, 40, 39]. Although there has been some research that has shown better results when incorporated with the state of art VecMap model [40, 48], none has conducted research that combines multiple approaches such as combining an improved modified version of the VecMap model with post-processed embeddings or combined multiple improvement techniques.

Chapter 3

METHODOLOGY

3.1 Overview

We have identified that only a limited number of research has been carried out for LRLs on BLI. Additionally, no previous research or comprehensive analysis has been conducted in terms of the size and the nature of the dataset used to create the word embedding model, the type of the evaluation dictionary used to evaluate the results, or the approach that has been used to create the word embeddings for BLI.

In this research, all the experiments were carried out on low-resource Sinhala-English language pair. We experimented with three different recently introduced BLI models VecMap, InstaMap, and ClassyMap for the Sinhala-English pair to identify the best baseline model to proceed with further experiments. We decided to move forward with the VecMap model due to reasons which will be further discussed in the coming sections. We conducted an extensive set of experiments on how the size and the nature of the dataset, the type of the evaluation dictionary and the approach used to create the embedding models affect BLI using the VecMap model.

We further introduced multiple post-processing steps to the pretrained embeddings used in the VecMap model before performing the alignment/mapping of the embeddings of each language.

Finally, we have combined the post-processing of the embedding spaces/linearly transforming the embedding spaces with the modified VecMap to obtain vastly superior results. The research process can be seen in Figure 3.1.

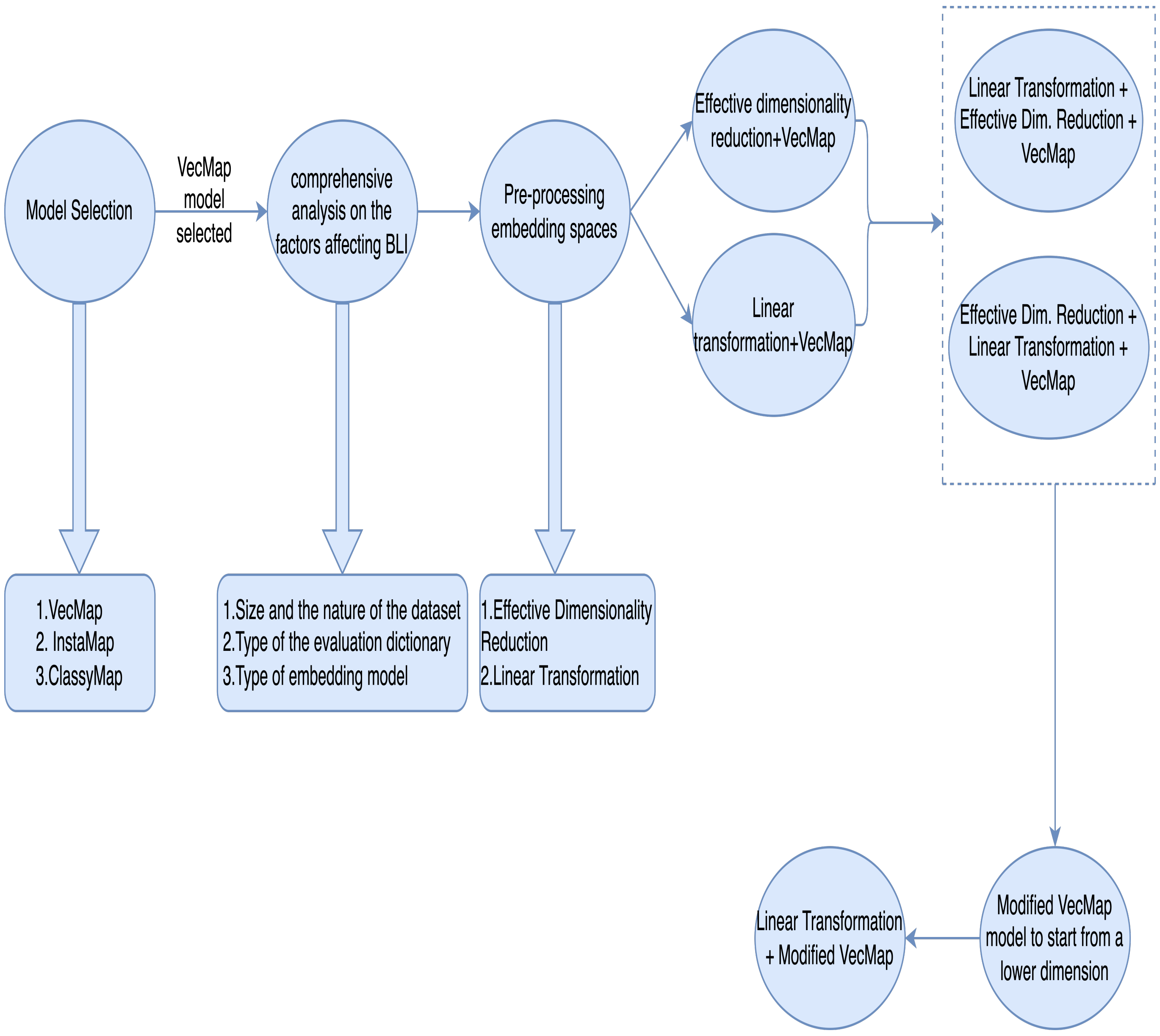

Model Selection
VecMap model selected
comprehensive analysis on the factors affecting BLI
Pre-processing embedding spaces
Effective dimensionality reduction+VecMap
Linear transformation+VecMap
Linear Transformation + Effective Dim. Reduction + VecMap
Effective Dim. Reduction + Linear Transformation + VecMap
1.VecMap
2. InstaMap
3.ClassyMap
1.Size and the nature of the dataset
2.Type of the evaluation dictionary
3.Type of embedding model
1.Effective Dimensionality Reduction
2.Linear Transformation
Modified VecMap model to start from a lower dimension
Linear Transformation + Modified VecMap

Figure 3.1: Research Process

3.2 Model Selection

We have selected three different models to run our experiments namely VecMap, InstaMap, and ClassyMap models. All selected models find a mapping between the embedding spaces of two different languages using monolingual embeddings of each language. Sinhala being a Low resource language, it is challenging to find a large amount of parallel data or a sizable seed dictionary to provide supervision with any other language. Since each of the selected models reported results on par with other existing models for BLI and since the required resources can be provided even for low-resource languages, we have selected these three models as the best models to conduct initial experiments on BLI for Sinhala and English languages.

3.2.1 VecMap Model

The VecMap model finds considerably accurate mapping between two languages using monolingual embeddings of each language. As discussed in 2.6.4 VecMap model consists of four sub-steps which are Pre-processing, Initialization, Robust self-learning, and Symmetric re-weighting (Figure 2.5). We have selected the fully unsupervised version of the VecMap model due to the unavailability of a large seed dictionary and since the unsupervised version reported the best result for all the preliminary experiments conducted.

3.2.2 InstaMap Model

The InstaMap model consists of two main steps;

1. **Finding a globally optimal rotation:** InstaMap model first finds a globally optimal rotation using the kabsch algorithm for the source embedding space w.r.t target embedding space. In this step, Glavavs et al. [48] used the whole training dictionary to find and applied PCA without dimensionality reduction to find the globally optimal rotation.

2. **Obtaining instance-specific translations:** An instant-specific translation vector for each point is calculated. This is calculated using the nearest neighbours' translation vector for the relevant point in the training dictionary. Then each point is moved along the aforementioned calculated translation vector.

This research showed that it can obtain better results when the proposed model was applied on top of the VecMap model. Additionally, the model reported accurate results when using a seed dictionary/training dictionary between the size of 5k-10k word translation pairs [48]. Sinhala being a low-resource language, a dictionary of such size was unavailable and difficult to produce. Hence we conducted our initial experiments between about 500-800 word translation pairs. This model reported very low results for Sinhala and English languages compared to the VecMap model since the size of the evaluation dictionary is considerably low and therefore is unable to find an accurate mapping between the embedding spaces.

3.2.3 ClassyMap Model

ClassyMap model incorporates classification-based self-learning with iterative mapping of cross-lingual embedding spaces. This approach enables the integration of various features into the iterative mechanism of the mapping. This allows the model to capture different levels of features such as orthographic level or word level. As this makes way to capture information like sub-wording information, this approach seemed attractive for Sinhala and English, since the Sinhala language is a morphologically rich language with a heavy inflectional nature. Furthermore, this approach reduces the noise, since each iteration method allows to check the reliability of the candidate translation pairs considered for that iteration.
This method also showed that it is able to obtain better results when integrated with the VecMap model and only required monolingual embeddings of source and target language with about 500-word translation pairs as the seed dictionary. Karan et al. [40] use supervised classification in the classification step of

this method. However, results obtained by applying the ClassyMap model for Sinhala and English language pair were unable to surpass the results obtained by applying the unsupervised version of the VecMap model.

3.2.4 Summary

We have run preliminary experiments for all three models using the same dataset and the seed dictionary (when applicable). As the unsupervised version of the VecMap model reported Superior results for Sinhala and English language pairs from all three models, we decided to proceed with the unsupervised version of the VecMap model for all future experiments.

3.3 Extensive analysis on BLI in low resource language pairs

We have conducted a comprehensive analysis of how the following factors affect BLI.

- Size and the nature of the monolingual data used to train the embeddings
- Properties of the bilingual dictionaries used for evaluation
- Type of the model used to train the embeddings

Each of the above items is discussed below.

3.3.1 Size and the nature of monolingual data

We have trained several different monolingual embedding types for both Sinhala and English languages.

1. **Pre-trained monolingual embedding models:** For both Sinhala and English languages, there are pretrained sets of monolingual embeddings that were readily available[1]. We have conducted our preliminary experiments using these embeddings for the VecMap model. We next proceeded to use Sinhala and English comparable data.

[1]https://fastText.cc

2. **Training embedding models using Sinhala and English comparable news data:** As news data can be considered to be an excellent source to extract comparable Sinhala and English data, we used news data to train monolingual embeddings. We have used several different news sources available such as Hiru, NewsFirst, Army, LankaPuwath, etc that were pre-extracted [49]. We pre-processed each news source separately for each language. We removed special characters and punctuation from the corpora of both languages. We further pre-processed the Sinhala corpora to remove characters from other languages. We combined(appended together) the pre-processed corpus of every news source of each language separately to generate a large monolingual corpus for both source and target languages. We then trained the large corpora to obtain monolingual embeddings for both languages. We trained both Word2Vec and FastText embedding models for both languages. Then this large monolingual dataset was used to find a cross-lingual mapping between Sinhala and English languages using VecMap, InstaMap, and ClassyMap models.

   Since the data become more comparable, if monolingual embedding spaces of each news source were mapped separately, we next trained embeddings for each news source for each language using both fastText and Word2Vec Models. Upon manual inspection, we observed that different news sources had different writing styles. To further analyze this we have created corpora for each language

   (a) by combining the data of similar writing patterns(Hiru news data and NewsFirst news data)

   (b) by combining the data of different writing patterns(NewsFirst news data and Army news data)

   We then created both FastText and Word2Vec monolingual embeddings for those corpora. Next, we proceeded to find a mapping for each type using the VecMap model.

   Since NewsFirst data reported the best results, we proceeded to double the size of the NewsFirst dataset by collecting more data to further monitor how the size of the dataset affects BLI. We then trained monolingual embeddings for each language and applied the VecMap model to observe how the size of the dataset affects BLI.

3.3.2 Type of The Evaluation Dictionary

Typically, BLI is evaluated by either integrating a BLI system with a task-specific downstream system such as neural machine translation or named entity recognition to obtain the resulting improvement of an entire system or by evaluating the results against a golden standard dictionary. In this research, we proceeded to evaluate the results against a golden standard dictionary. We evaluated the results of our experiments against a few different dictionaries. We proceeded to evaluate the results with multiple evaluation dictionaries to observe how the results change and how the nature of the evaluation dictionary affects the results.

1. **Standard Dictionary**- We evaluated the results of our experiments initially using an existing standard dictionary [50].

2. **Term Frequency dictionaries**- To create these dictionaries, using TF-IDF, we calculated the term frequencies of each word for both languages using combined news corpora. We then created

   - High-frequency term dictionary - Word translation pairs were obtained by using the terms that appeared as high-frequent terms in both languages.
   - Low-frequency term dictionary - Word translation pairs were obtained by using the terms that appeared as low-frequent terms in both languages.

3. **Dictionary retrieved from parallel data**- To differentiate the dictionary from the dataset, we have created another dictionary using a dataset differing from the original dataset used to create the embeddings and is from a different domain to the news domain. For this, we used parallel data from

the government document domain, which is extracted from Official Government Documents [50]. We have again taken the term frequencies for each language for parallel data and taken the high-frequency word translation pairs of each language to create the dictionary.

3.4 Post-processing the Embedding Spaces

As the next step of our research, we decided to post-process the pre-trained embeddings generated, to obtain a better mapping between two languages. In this step, we have selected two different approaches to post-process the pre-trained embeddings from the mentioned post-processing approaches in subsection 2.7.

1. Linear Transformation
2. Effective Dimensionality Reduction

3.4.1 Linear transformation

This post-processing approach is able to capture the information that is not immediately apparent in pretrained embeddings and, by post-processing the embedding, post-processed embeddings will cater to the application the embeddings are being used. Also, it has been empirically validated that post-processing pre-trained embeddings before obtaining a cross-lingual word embedding model is beneficial for low resource BLI [22]. Additionally, this approach only requires pretrained monolingual embeddings to experiment with. Therefore we selected this approach as our first post-processing technique.

Conventional projection-based cross-lingual mapping approaches independently train source language(L_s) monolingual embeddings X and target language(L_t) monolingual embeddings Z separately and learn a mapping between two languages applying a linear transformation using a seed dictionary D. The initial mapping is then improved iteratively until the algorithm converge. In an unsupervised scenario, this dictionary is derived in the first iteration of the mapping approach.

We experimented with the linear transformation approach for post-processing pre-trained embeddings introduced by Artetxe et al. [21] in this research. The essence of this approach is to generalize the idea of first-order similarity, second-order similarity up to n-th order similarity. If the standard first-order similarity matrix of source embedding matrix X is $M_1(X) = XX^T$ then the second order similarity for source X can be defined as $M_2(X) = XX^TXX^T$. Hence $M_2(X) = M_1(M_1(X))$. Accordingly the n-th Similarity order of X can be defined as $M_n(X) = (XX^T)^n$. It is proven by Artetxe et al. [21] that this n-th order similarity transformation can be also obtained using $M_n(X) = M_1(XR_{(n-1)/2})$ where $R_\alpha = Q\Delta^\alpha$. Here Q and Δ are obtained by the eigen decomposition of $XX^T = Q\Delta Q^T$ where Q is the orthogonal matrix with eigen vectors and Δ is the diagonal matrix containing the eigen values. We then obtain the linearly transformed embedding spaces of source embeddings(X) as $X^l_{\alpha_s} = XR_{\alpha_s}$ and linearly transformed target embeddings(Z) as $Z^l_{\alpha_s} = ZR_{\alpha_s}$. Finally before attaining the cross-lingual word embedding mapping between two embedding spaces we replace the source embedding(X) and target embedding(Z) by $X^l_{\alpha_s}$ and $Z^l_{\alpha_s}$ consecutively.

3.4.2 Dimensionality reduction

Almost all the available BLI techniques have been experimented with in comparatively high dimensions such as 300 and usually, the experiment was done for a very large number of tokens. This increases the memory requirement of the models. Furthermore, the required training time is also comparatively high for such models. In this research, as we conduct an extensive analysis with a large number of experiments, the importance of an efficient model was apparent. Hence to increase the performance without compromising the results, we have introduced effective dimensionality reduction with a post-processing algorithm(Algorithm 1) to increase the results of our model as the next post-processing step of the embeddings. We followed the research steps of- Raunak et al. [23] in this part of our research as guidance.

The Algorithm 1 is based on the assumption/observation across all the embed-

Algorithm 1: Post Processing Algorithm (X,D)

```
  Input: Word embedding matrix X, Threshold parameter D
  Output: Post-processed Word embedding matrix X
  /* Subtract the mean embedding */
1 X = X - $\overline{X}$
  /* Compute PCA components */
2 $u_i$ = PCA(X) ; i= 1, 2, 3, ....d;
  /* Remove top D components */
3 for all $v$ in $X$ do
4     $v = v - \sum_{i=1}^{D}(u_i^T.v)u_i$
5 end;
```

dings models, that the embeddings consist of a large mean vector, and once the mean vector is subtracted from the original embedding the energy of the remaining embedding is spanned in a subspace consisting about eight dimensions. Since the mean vector is common amongst all embeddings, by eliminating the mean vector, embeddings become stronger. This makes the embeddings more discriminative since, in the newly obtained embeddings, none of the principal components are disproportionately dominant securing better quality embeddings [46].
Following Raunak et al. [23], to obtain the discriminative embeddings, first the Algorithm 1 is applied to the original set of embeddings. Then on top of these purified sets of embeddings dimensionality reduction was done using the PCA dimensionality reduction technique. Finally, to further flatten the dominant components of the dimensionality reduced embeddings we again applied Algorithm 1 on the dimensionality reduced embeddings.

3.4.3 Linear transformation with dimensionality reduction

As both the linear transformation and dimensionality reduction deliver different benefits to pretrained embeddings and to leverage from both post-processing techniques, we decide to incorporate both techniques into our model. We first experimented with linear transformation with a few different values for α hyperparameter for both source and target monolingual embeddings. We selected best suited α value we can benefit from when incorporated with the VecMap algorithm.

Next, we integrated the effective dimensionality reduction model with the VecMap algorithm for a half-sized dimension incorporated with the post-processing algorithm.

Finally, to benefit from both techniques, we introduced two hybrid models where we follow the steps, first applying the linear transformation, then the effective dimensionality reduction on the pretrained monolingual embeddings before applying the VecMap model to obtain a mapped embedding space. To observe the difference in order these post-processing techniques were applied, we then proceeded to experiment first applying the dimensionality reduction on the pretrained embeddings and then applying the linear transformation to finally apply the VecMap algorithm to obtain the cross-lingual word embeddings.

3.5 Iterative Dimensionality Increment With VecMap

Sinhala being a LRL, resources including large enough parallel data set or bilingual dictionaries with few thousand words is considered rare and almost impossible to obtain without extensive and expensive manual intervention. Hence for tasks such as BLI, it is not possible to pick supervised or semi-supervised approaches as LRLs lack the required resources. Therefore most of the LRLs heavily rely upon unsupervised approaches for BLI.

When obtaining the cross-lingual word embeddings using fully unsupervised approaches, these approaches self initialize the initial seed dictionary and iteratively improve the solution. The self-initialization step is an important part of these methods as the converged solution heavily depends on this self-initialization step. Li et al. [51] showed that if the initialization step is not accurate enough self-learning will not converge to a solution. This will cause the accuracy of the results to be zero or closer to zero.

We first use PCA to reduce the dimensionality of the initial raw word embeddings as this will reduce the dimensions that are less important when explaining the core features of the data. Therefore this can be considered as a *noise-reduced* *embedding space.* Hence if an initial solution is to be obtained using this noiseless dimensionality reduced embeddings the initialization tends to be rather accurate. Also, to avoid the hubness problem [52, 51], drop-max is applied along with the PCA algorithm. Following Li et al. [51], we first iteratively reduce the dimension of the raw embeddings till they reach the dimension of 50. We then apply the state-of-the-art VecMap algorithm on the dimensionality-reduced Embeddings to obtain the initial solution dictionary. Then using the initial dictionary algorithm will add k most frequent words to the initial dictionary calculating nearest neighbours. Then the dimensionality of the embeddings was doubled to repeat the process where the resultant dictionary of the previous step will act as the seed dictionary. These steps will repeat until the dimensionality of the embeddings is less than or equal to the original dimensionality of the initial raw embeddings (Refer Algorithm 2).

Algorithm 2: ITERATIVE DIMENSIONALITY INCREMENT (E,n)

```
   /* Initial dimensionality of the Raw embeddings = 300      */
   Input: Raw word embedding matrix E, Initial dimension n
   Output: cross lingual word embeddings of source and target
           embeddings $W_X$ and $W_Z$ respectively
1  D ← $\phi$
2  while $n <= 300$ do
3      Reduce E to $\overline{E}$ with dimension min(n,300) using PCA and dropmax
4      if $D = \phi$ then then
5          Run the initialization and self-learning on $\overline{E}$
6      else
7          Run the self-learning on $\overline{E}$ with D as an initial dictionary
8      end if
9      Translate 4k most frequent words and store the results in D
10     n ← $n * 2$
11 end while
12 return $W_X$ and $W_Z$;
```

3.6 Modified VecMap Model With Pre-processed Embeddings

As a final step to our model, we have integrated the iterative dimensionality reduction of the VecMap algorithm with the post-processing of raw embeddings.

As per the experiments conducted in subsection 3.4.3 we were able to obtain improvement for each pre-processing approach when individually integrated with the VecMap algorithm. Yet when both preprocessing approaches were combined and integrated with the VecMap model we were unable to obtain substantial gains. Hence to proceed further we selected the post-processing approach that yielded the best results when combined with the initial VecMap algorithm which is the linear transformation approach. We then proceeded to integrate the linear transformation post-processing approach with the modified version of the VecMap algorithm to gain the all-time highest results of BLI in Sinhala and English Languages.

Chapter 4

EXPERIMENTS

4.1 Experimental Setup

For the initial experiments, we used the Google Colab pro version with P100 and T4 with high-memory virtual machines. As the dataset grew, the memory consumption increased and hence we conducted the rest of the experiments on RTX 6000 NVIDIA GPU machine with 64GB RAM.

4.2 Data

4.2.1 Corpora and Embeddings

For the preliminary level experiments, we used the existing pretrained embedding for both Sinhala and English languages[1]. This dataset consisted of 200k embeddings with a dimension of 300. Since this gave us considerably diminished results, we decided to experiment with comparable data. As Sinhala and English news data can be considered excellent sources for comparable data we then proceeded to experiment with Sinhala and English data from multiple news sources extracted by Rajitha et al. [49]. We then pre-processed the data as mentioned in subsection 3.3.1. Then we manipulated the cleaned data to obtain several different types of datasets(stats are at Table 5.1).

1. We trained both Word2Vec and FastText monolingual embeddings for each news source for both Sinhala and English languages

2. Trained both Word2Vec and FastText monolingual embeddings for English and Sinhala combining all the news sources for each language.

3. Created separate corpora for each language by combining news sources that

[1]https://fastText.cc

had similar writing styles (NewsFirst+Hiru) and trained both word2vec and fastText monolingual embeddings for each language

4. Created separate corpora combining news sources that had different writing styles (NewsFirst+Army) and trained both word2vec and fastText monolingual embeddings for each language

5. As news first news source has given the best results when applied VecMap, we have crawled more news first data to increase the corpora size (approximately doubled the size of the news corpora) and trained both fastText and word2vec embeddings for this larger corpora.

All the mentioned embedding models trained were of dimension 300.

Table 4.1: Dataset

Dataset	**Si Embeddings**	**En Embeddings**
Pre-trained emb.	200,000	200,000
Combined news	153,431	162,301
NewsFirst	64,535	75,438
Hiru	56,045	58,545
Army	36,243	43,245
NewsFirst+Hiru	72,043	79,635
NewsFirst+Army	65,344	79,326

4.2.2 Evaluation dictionaries

We have evaluated the results against a few different types of evaluation dictionaries to observe how the results of BLI depend on the nature of the evaluation dictionary. The dictionaries we used are as follows.

1. Standard dictionary - This is an existing standard Sinhala and English dictionary consisted about 36,000-word translation pairs between Sinhala and English languages [50].

2. High-frequency dictionary - Dictionary consisted of high-frequency word translation pairs that appeared in our dataset(subsection 3.3.2). This dictionary consisted of about 500-word translation pairs.

3. Low-frequency dictionary - Dictionary consisted of low-frequency word translation pairs that appeared in our dataset(subsection 3.3.2). This dictionary consisted of about 500-word translation pairs.

4. Parallel corpus dictionary - Dictionary created using Sinhala-English parallel dataset. This dataset consisted of 74,000 Sinhala-English parallel sentences. This dictionary also consisted of about 500-word translation pairs.(subsection 3.3.2)

4.3 Experiments

4.3.1 Comprehensive analysis

We first trained the VecMap model to obtain cross-lingual word embeddings for Sinhala and English languages using an existing pre-trained monolingual embedding set and evaluated the results against the standard dictionary.
We then experimented with Sinhala English comparable news data. We trained both fastText and word2vec embeddings for each corpora type mentioned in subsection 4.2.1. We then applied the VecMap algorithm for the embeddings created using corpora created by combining all news sources. We evaluated the results against the Standard dictionary, high-frequency dictionary, and low-frequency dictionary. We then applied the VecMap algorithm for the embeddings created using each news source separately. Then we evaluated the results using the high-frequency dictionary and low-frequency dictionary. As the NewsFirst news source presented the best results thus far to further validate the results, we evaluated the NewsFirst cross-lingual embedding results against the parallel data dictionary next. We then doubled the size of NewsFirst corpora and trained monolingual embeddings using both fastText and word2vec models and applied the VecMap model. Evaluated the obtained cross-lingual embedding model against the high-

frequency dictionary, low-frequency dictionary, and parallel data dictionary. Upon manual inspection, we noticed that some news sources had different writing styles while some of the news sources had similar writing styles. Hence we have created two different corpora by combining two similar writing-styled corpora and two different writing-styled corpora. we then proceeded to train both fastText and word2vec embeddings for the aforementioned corpora and applied the VecMap algorithm. We evaluated the results against high-frequency dictionaries and parallel data dictionary.

4.3.2 Post-processing pretrained embeddings

Since the NewsFirst news source dataset gave the best results across all news sources when evaluated against all types of dictionaries we selected the NewsFirst dataset for the next part of the experiment set.

1. **Linear transformation:** In this step we post-process the source pretrained embedding vector X to $X^{/}_{\alpha_s}$ and Target pretrained embedding vector Y to $Y^{/}_{\alpha_t}$. This requires tuning two hyperparameters α_s and α_t for source and target languages respectively. As a consequence of lacking the necessary resources to tune the hyperparameters, we manually specified a subset of the hyperparameter space based on the experiments conducted by Vulic et al. [22]. We post-processed the NewsFirst dataset for both source and target languages for the hyperparameter values [-0.5, -0.25, -0.15, 0, 0.15, 0.25, 0.5]. We then obtained cross-lingual word embeddings by applying the VecMap model to each value pair for both languages and evaluated the results against the parallel data dictionary. This experiment was conducted for both fastText and word2vec embeddings.

2. **Effective dimensionality reduction:** We proceeded to apply dimensionality reduction with post-processing algorithm 3.4.3 as the next step of our research. We started the dimension of our original embeddings as 300. We then halve the size of the dimension for both source and target pretrained embeddings to obtain embeddings with 150 dimensions as the

post-processed embedding set following the steps of Raunak et al. [23]. We then obtained cross-lingual word embeddings for dimensionality-reduced word vectors by applying the VecMap algorithm and evaluated the results against a parallel data dictionary. This experiment is conducted for both fastText and word2vec embeddings.

4.3.3 Post-processing with modified VecMap model

As the next step of our research, we selected the best suitable hyperparameters based on the results of section 4.3.2. For fastText embeddings hyperparameter values selected were $\alpha_s = 0.15$ and $\alpha_t = 0.5$ while the hyperparameter values selected for word2vec model is $\alpha_s = 0.15$ and $\alpha_t = -0.15$. We then applied effective dimensionality reduction for these the post-processed embeddings using the aforementioned hyper-parameter values. We also applied linear transformation on a dimensionality-reduced embedding set. Next, we applied the VecMap algorithm to both post-processed embedding sets. We evaluated the results using the parallel data dictionary. As combining both post-processing techniques diminished our results, for the next part of our experiments we decided to only apply the technique that improved the results best. Hence we selected linear transformation as the post-processing technique for the latter part of our experiments. Next, we improved the VecMap algorithm by iteratively increasing the dimensionality of the embeddings(Algorithm 2). Evaluated the results of the improved VecMap model using the parallel data dictionary. Finally, we further improved the results by applying the post-processing technique linear transformation, on the pretrained embeddings before applying the improved VecMap algorithm.

Chapter 5

RESULTS AND DISCUSSION

All the results for experiments conducted were reported by precision@K according to the standard practice. precision@K is the number of correct translations in the top k set out of the retrieved result set and we report the results when k equals 1.

5.1 Comprehensive Analysis

5.1.1 Pre-trained fastText embeddings

Table 5.1: BLI Results For Pre-trained fastText Embeddings

Dictionary	**fastText P@1%**
Standard Dic.	0.02
Parallel data Dic.	1.04

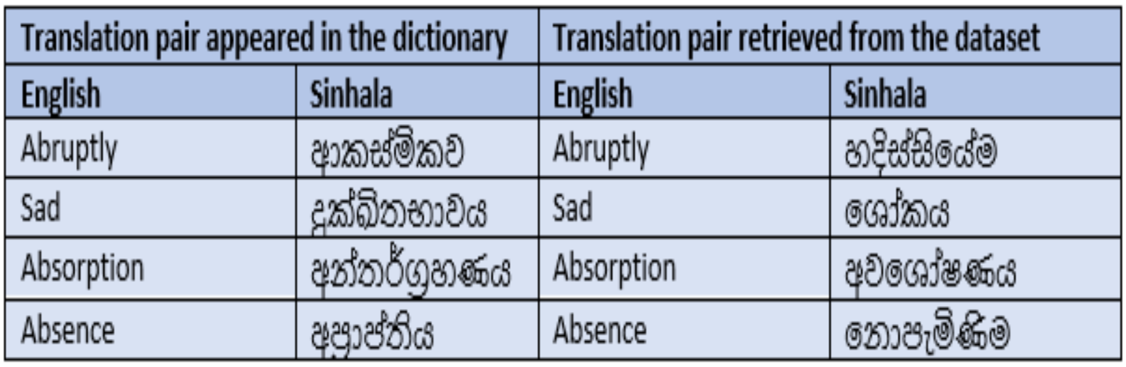
The results of the pre-trained embedding were evaluated against the standard existing dictionary and the parallel dictionary and can be seen in Table 5.1. As can be observed, when evaluated against the standard existing dictionary the result for the BLI is considerably low. However, the results were improved even though the result is still extremely poor when evaluated against the parallel data dictionary as an ablation test.
Even though the standard dictionary consisted of a considerably larger number of word translation pairs in comparison to the parallel data dictionary, the Sinhala word translation in the standard dictionaries was rarely used every day as a practice. This phenomenon is further discussed in the next subsection (subsection 5.1.2.)

5.1.2 Combined News Data

Table 5.2: BLI Results For Combined News Data

Dictionary	**Word2Vec P@1%**	**fastText P@1%**
Standard Dic.	0.60	0.43
High frequent term Dic.	12.14	8.05
Parallel data Dic.	8.73	7.87

It is apparent that the results (Table 5.2) for BLI improve dramatically when the dataset is at least slightly comparable. Also, it is observable that the results improve drastically when evaluated against the high-frequent term dictionary and parallel data dictionary. Theoretically, as the standard dictionary consists of a large number of translation pairs compared to the other two dictionaries used, the probability of retrieving a correct translation pair to match from the standard dictionary should be higher. However, upon manual inspection, it is apparent that the translation pairs that appeared in the standard dictionary are rarely used in everyday use. Examples can be seen in the Figure 5.1.



Translation pair appeared in the dictionary		Translation pair retrieved from the dataset	
English	Sinhala	English	Sinhala
Abruptly	ආකස්මිකව	Abruptly	හදිස්සියේම
Sad	දුක්ඛිතභාවය	Sad	ශෝකය
Absorption	අන්තර්ග්‍රහණය	Absorption	අවශෝෂණය
Absence	අප්‍රාප්තිය	Absence	නොපැමිණීම

Figure 5.1: Translation Difference Examples

Similar to the example most of the words that appear in the dictionary do not appear in the dataset. Hence, even when the correct translation is retrieved, when evaluated against the standard dictionary the translation will be taken as incorrect reducing the accuracy of the results. Also, it can be seen that the re-

sults increase drastically when evaluated against the high-frequency dictionary. In view of the fact that the word translation pairs in the high-frequency dictionary appear in the dataset used, is the cause for higher results.
In addition, it can be seen that the results, when evaluated against the parallel data dictionary, are slightly lower than the results evaluated against the high-frequency dictionary, although it is still significantly greater than when evaluated against the standard dictionary. The parallel dictionary is created from the government documents, which is an entirely different domain from the dataset domain news. Therefore it can be argued that the frequent word translation pairs that appear in the parallel dictionary are different to some extent from the news domain. However, the translation pairs that appeared in the parallel dictionary are used in everyday applications and therefore give significantly better results in comparison with those evaluated against the standard dictionary. In conclusion, it is evident that the nature of the evaluation dictionary highly impacts the results of BLI.

We also made the observation that word2vec results were slightly better throughout when evaluated against all the dictionaries. This phenomenon is further discussed in the next subsection 5.1.3.

5.1.3 Separate news sources

Table 5.3: BLI Results For Each News Source Separatelyfit to pg width

Dictionary	**Word2Vec P@1(%)**			**fastText P@1(%)**		
	NewsFirst	***Hiru***	***Army***	***NewsFirst***	***Hiru***	***Army***
High frequent term Dic.	42.57	40.01	27.55	40.33	37.75	27.79
Low frequent term Dic.	35.34	32.27	27.48	40.94	36.15	28.38
Parallel data Dic.	28.47	29.58	20.59	27.78	30.63	24.77

The news data that was extracted for each news source for both languages are in the same time frame (i.e., monolingual NewsFirst data collected for both Sinhala and English is in the same time frame, and similarly, Hiru news data extracted for both Sinhala and English is in the same time frame). Hence it is safe to assume that the news articles in the two languages will roughly discuss the same topics. Therefore monolingual corpora from separate news sources will be more comparable compared to the corpora built by combining all the news sources together. Therefore to survey how the nature of the data affects the results of BLI, we evaluate the BLI results for each news source separately against the high-frequency dictionary, low-frequency dictionary, and the parallel dictionary. As reflected in Table 5.3, we were able to gain substantial gains on the result set when evaluated against all three dictionaries. The highest results we were able to gain were when evaluated against high-frequency dictionary for NewsFirst news source for word2vec model and the lowest results were obtained was when evaluated against parallel data dictionary for Army news source for word2vec model.

Also when evaluated against the high-frequency dictionary, results were always higher or slightly different for the word2vec embedding model. However, when evaluated against the low-frequency dictionary, results were always higher for the fastText embedding model. We noticed that most of the terms that appeared in the low-frequency dictionary were rare terms which are either inflected terms of high-frequency terms or named entities. Since the fastText model better accommodates sub-wording information in comparison to the word2vec model fastText model was able to acquire a better mapping for inflected terms. Therefore fastText model was able to obtain better translations for rare inflected terms. Hence it can be concluded that a fastText embedding model is a superior option in respect of BLI for rare terms.

Also, it can be observed in Table 5.3 that the results were slightly dropped when evaluated against the parallel data dictionary throughout all news sources and embedding types. As discussed in the subsection 5.1.2 parallel dictionary has a relatively different domain compared to the dataset and the high-frequency and low-frequency dictionaries having the same domain as the dataset is the cause of this.

As an ablation test and to observe how the size of the dataset affects the results, we increased the size of the NewsFirst dataset by crawling more data. We then proceeded to clean and train monolingual embeddings for this large NewsFirst dataset. We then performed the VecMap algorithm to obtain cross-lingual word embeddings for both fastText and VecMap embedding models. The results for the fastText embedding model increased to 31.77 from 27.78 resulting in a gain of 5.99, whereas results of the word2vec embedding model were improved to 33.75 from 28.47 causing an increment of 5.28 when evaluated against the parallel data dictionary. As it can be observed that the size of the dataset is largely increased the results were improved albeit by about 4 percent for both embedding models. Hence it is evident that the nature and the size of the dataset directly impact the results of BLI.

5.1.4 Combined news data based on writing styles

Upon manual inspection, it was apparent that each news source had a unique writing style concerning the sentence styles, the vocabulary used, etc and it can be observed that some of the news sources had similar writing styles while some news sources had completely different writing styles.
To further observe how the nature of the dataset affect BLI, we created separate corpora combining two similar writing-styled news sources(Hiru+NewsFirst) and two different writing-styled news source(Army+NewsFirst). As it can be observed in Table 5.4 and Table 5.3 when the similar writing-styled news sources were combined(NewsFirst+Hiru) results were slightly diminished lower than NewsFirst data, yet the results remained higher than the Hiru news data when evaluated against frequency dictionaries. Additionally, when evaluated against the parallel dictionary the results dropped trivially lower for the combined Hiru+NewsFirst dataset than the results of Hiru or NewsFirst data alone.

However, as can be observed from Table 5.4 and Table 5.3 when different writing-styled news sources were combined (NewsFirst+Army), the results dropped substantially lower than the results of either NewsFirst or Army news data alone when evaluated against all three dictionary types. By combining dif-

Table 5.4: BLI Results For Few News Sources Combined

Dictionary	**Word2Vec P@1(%)**		**fastText P@1(%)**	
	NewsFirst+ Hiru	***NewsFirst+ Army***	***NewsFirst+ Hiru***	***NewsFirst+ Army***
High-frequency Dic.	40.98	17.38	38.85	16.02
Low-frequency Dic.	33.68	14.21	36.98	17.44
Parallel data Dic.	28.26	10.56	26.87	13.67

ferent news source data, the comparability of the source corpus and target corpus will reduce causing a drop in the results. Furthermore, the embedding models used for the experiments consider contextual information to build the embedding space. It can be assumed that when the context drastically changes in the corpora, finding a mapping between the created embedding spaces for source and target could be increasingly difficult. Thus, the results of this experiment further reinforce the conclusion that the nature of the dataset used for BLI is highly relevant and affect the results of BLI.

5.2 Post-processing Pre Trained Embeddings

This set of experiments was conducted for the large NewsFirst dataset and all the results were evaluated against the parallel dictionary. The parallel dictionary is used since the dataset used is different from the dataset used to create the embedding models and hence can be considered as a more accurate means to evaluate the results.

5.2.1 Linear transformation

As discussed in the subsection 4.3.2 we first post-processed the pretrained embeddings using the linear transformation technique for both fastText and word2vec embedding models for both Sinhala and English languages for a few different hyper-parameter values. The results for fastText embeddings can be seen in Table 5.5 and the results for the word2vec embedding model are demonstrated in Table 5.6. Although several hyper-parameter values were able to improve the existing results the maximum increment over the results was obtained for the

Table 5.5: BLI Results For Linear Transformation fastText Embeddings

Si / En	**-0.5**	**-0.25**	**-0.15**	**0**	**0.15**	**0.25**	**0.5**
-0.5	0.00	0.00	0.00	0.00	0.00	0.00	0.00
-0.25.	0.00	32.44	31.77	32.44	2.34	0.33	0.33
-0.15	0.00	0.33	33.44	1.67	32.78	0.67	1.67
0	0.00	6.69	0.67	31.77	34.11	34.78	33.44
0.15	0.00	2.01	30.77	0.67	34.78	33.44	**35.12**
0.25	0.00	0.00	31.44	32.44	33.44	33.11	34.11
0.5	0.00	0.33	0.00	1.00	0.00	29.43	25.75

Table 5.6: BLI Results For Linear Transformation word2vec Embeddings

Si / En	**-0.5**	**-0.25**	**-0.15**	**0**	**0.15**	**0.25**	**0.5**
-0.5	0.00	0.00	0.00	0.00	0.00	0.00	0.00
-0.25.	0.32	0.32	2.84	32.49	29.02	30.28	25.87
-0.15	0.32	0.63	33.12	33.12	0.00	0.63	26.50
0	0.00	32.49	0.00	33.75	32.81	31.55	25.24
0.15	0.00	33.75	**34.87**	32.81	33.44	0.32	25.55
0.25	0.00	32.49	33.12	31.86	32.81	32.18	26.50
0.5	0.00	0.00	0.00	30.28	29.65	29.97	23.97

fastText model when the $\alpha_s = 0.15$ and $\alpha_t = 0.5$. For the aforementioned hyperparameter values, we were able to improve the results to 35.12% from the 31.77% for the fastText model when evaluated against the parallel data dictionary. Similarly we were able to obtain improved results for word2vec embeddings for several hyperparameter values as can be observed in the Table 5.6. However the maximum improvement was obtained when $\alpha_s = 0.15$ and $\alpha_t = -0.15$. We were able to improve the results for the word2vec embedding model from 33.75% to 34.87%. It can be observed that the improvement gained for the fastText embedding model was far more significant compared to the improvement gained for the word2vec embedding model using the linear transformation model.

5.2.2 Effective dimensionality reduction

As already mentioned in the subsection 4.3.2, we next post-processed the raw pre-trained embeddings using effective dimensionality reduction with the post-processing algorithm 1. As this technique caters to efficiency without compromising the results we were able to obtain the results considerably faster. However, the improvement in the results for both models was diminutive. In relation to fastText embeddings, the results were improved to 32.02% from 31.77% when applied dimensionality reduction with the post-processing algorithm. The results were increased from 33.75% to 33.86% for the word2vec embedding model when applied dimensionality reduction with the post-processing algorithm. The results were evaluated against the parallel data dictionary.

5.2.3 Linear transformation + Effective dimensionality reduction

Next, we combined two post-processing methods before obtaining cross-lingual word embeddings by applying the VecMap algorithm. In this step, we used the hyperparameter values that gave the best results based on the experiments done in subsection 5.2.1. We first applied effective dimensionality reduction on linearly transformed embeddings for both fastText and word2vec embedding models. We reduced the dimensionality by half(to 150) for the linearly transformed word2vec

embeddings using the hyper-parameters $\alpha_s = 0.15$ and $\alpha_t = -0.15$ and for the linearly transformed fastText embeddings using the hyperparameters $\alpha_s = 0.15$ and $\alpha_t = 0.5$. The results obtained by this step are considerably lower compared to the original results where results for fastText embeddings diminished from 31.77% to 27.34% while word2vec results dropped from 33.75% to 28.32%.
As the results were reduced for the previous experiment, we tried to improve the results of dimensionality-reduced embeddings by applying linear transformation as a post-processing step to the low-dimensional embeddings. Although it has not been proven that the best hyper-parameters to use for these low dimensional embeddings obtained were the same as the ones we used for high dimensional embeddings with the limited time frame, we proceeded to use the above-obtained hyper-parameters that gave the best results for higher dimensions. Hence we used hyper-parameters $\alpha_s = 0.15$ and $\alpha_t = -0.15$ for the word2vec embeddings and hyperparameters $\alpha_s = 0.15$ and $\alpha_t = 0.5$ for the fastText embeddings. We were unable to gain improvement over the original results in this step again and the results were diminished compared to the original results. The results of word2vec embeddings dropped from 33.75% to 29.01%, while results for the fastText embeddings dropped to 28.46% from 31.77%.
In future work, we are expecting to run experiments for all the hyperparameter values to validate if the results can be improved with a different linear transformation with dimensionality reduction. We were unable to improve the results by combining two post-processing techniques in the scope of this research.

5.2.4 Improved VecMap model

As a further step, we introduced an improvement to the existing robust mapping algorithm VecMap. We introduced dimensionality increment in each iteration of the VecMap algorithm where the first iteration starts with a very low dimension to find a better initial solution(Algorithm 2). We were able to obtain significant improvements in the results by introducing this enhancement to the VecMap model for both embedding models. To ensure the resulting improvement we ran experiments for two different news sources NewsFirst and Hiru and then the results were evaluated against the parallel dictionary.

Table 5.7: BLI Results For Comparison for Iterative Dim. Reduction

	Word2Vec P@1(%)		**fastText P@1(%)**	
Algorithm	***NewsFirst***	***Hiru***	***NewsFirst***	***Hiru***
VecMap	33.75	29.58	31.77	30.63
Iterative dim. reduction with VecMap	35.77	31.21	34.96	32.44

As can be observed in Table 5.7, results were improved for both word2vec and fastText embedding models for both news sources. Therefore, it can be concluded that iterative dimensionality reduction improves the BLI results substantially across multiple embedding models for comparable monolingual data.

5.2.5 Post-processing pretrained embeddings+Improved VecMap model

Table 5.8: BLI Results For NewsFirst data in all steps

Algorithm	**Word2Vec**	**fastText**
VecMap	33.75	31.77
Effective Dim. reduction+VecMap	33.86	32.02
Linear transformation+VecMap	34.87	35.12
Linear transformation+Effective Dim. reduction+VecMap	28.32	27.34
Effective Dim. reduction+Linear transformation+VecMap	29.01	28.46
Iterative dim. reduction with VecMap	35.77	34.96
Linear transformation+Iterative dim. reduction with VecMap	37.89	38.06

As the next step of our research, we combined the post-processing pretrained embeddings with the enhanced VecMap algorithm to obtain better cross-lingual word embeddings. Since the linear transformation approach gave us better results compared to effective dimensionality reduction, we only used linear transformation as the primary post-processing method for pretrained embeddings. In this step, we first post-processed the pretrained embeddings of both fastText and word2vec embedding models for the hyperparameters that gave the best results when incorporated with the VecMap model. Hence hyper-parameter values $\alpha_s = 0.15$ and $\alpha_t = -0.15$ were used for word2vec embeddings while hyper-parameter values $\alpha_s = 0.15$ and $\alpha_t = 0.5$ were used with fastText embeddings.

In this step, we first post-processed the embeddings and then applied the improved VecMap algorithm with iterative dimensionality increment to the post-processed fastText and word2vec embeddings. The experiments were conducted for both Hiru and NewsFirst datasets. By incorporating these two techniques, we were able to gain substantial improvement for fastText and word2vec embedding models for both news sources used for the experiments when evaluated against the parallel data dictionary. Hence it is evident that this approach can improve the results of BLI substantially across different datasets and embedding models (table 5.8).

5.3 Summary

We have conducted an extensive set of experiments to identify the factors that affect BLI. We conducted experiments to find how the size and the nature of the dataset, the type of the evaluation dictionary, and the type of the embedding model used affect the results of BLI. Although existing research claims to find a mapping between low-resource languages using non-comparable corpora by applying the VecMap algorithm, we were unable to obtain a competitive result for Sinhala and English languages when using completely unrelated monolingual corpora. Results were improved drastically when used comparable corpora to create the embedding models. Also, results were very low for the existing standard dictionary due to the lack of use of the terms in the dictionary in the actual dataset. Also, it can be observed that the word2vec model was better when finding translations for frequent words while the fastText model is better when finding translations for rare terms.

We have also post-processed the pretrained embeddings using the techniques of effective dimensionality reduction and linear transformation. Applying linear transformation improved the results significantly in comparison to effective dimensionality reduction. Further, we improved the VecMap model by introducing incremental dimensionality reduction. Finally, we combined the post-processing of pretrained embeddings with the improved VecMap model to obtain even better results.

Chapter 6

CONCLUSION AND FUTURE WORK

Existing research for BLI has been dominantly conducted on HRLs. Only a handful of research was conducted for LRLs [53, 54, 55, 3]. None of the existing research conducted a comprehensive analysis of the factors that affect the results of BLI and scarcely any research focused on improving the existing approaches for BLI.

In this research, we conducted an extensive set of experiments w.r.t the factors, size, and the type of the dataset, type of the evaluation dictionary and the embedding model used, and how these factors affect the results of BLI. We conducted the experiments on the Sinhala-English low resource domain and used the existing robust VecMap model to obtain the cross-lingual word embeddings. Furthermore, we improved the obtained results by applying sophisticated post-processing techniques linear transformation and effective dimensionality reduction on top of pretrained embeddings. Additionally, we further improved the VecMap algorithm by introducing a novel technique of incremental dimensionality reduction in each iteration of the VecMap model. Finally, we further improved the results and introduced a novel technique for BLI by incorporating post-processing of pretrained embeddings with the improved VecMap algorithm.

It is evident that the size and the nature of the dataset, type of evaluation dictionary and the embedding model highly impact the results. It is apparent from the results that to obtain competitive results for LRLs similar to Sinhala dataset needs to have a substantial amount of comparable data. Furthermore, as the results for the BLI varied across different types of evaluation dictionaries used to evaluate, it can be concluded that the type of evaluation dictionary used can influence the results of BLI. Moreover, results concluded that word2vec embedding was able to find translation between frequent words whereas the fastText embedding model performed better when it comes to rare terms due to the fastText

models' capability to capture sub-wording information.
It was apparent that for LRLs such as Sinhala results of BLI can be significantly improved by applying suitable post-processing techniques on top of pretrained embeddings before obtaining cross-lingual word embeddings. Although, we were able to improve the results of BLI using the two different post-processing approaches effective dimensionality reduction and linear transformation, we were able to obtain substantial gains by applying the latter technique only. Also, as the improved VecMap algorithm delivered notable improvement over the VecMap algorithm across all the data when evaluated against different dictionaries it can be concluded that starting the algorithm in a lower dimension to get a better initial mapping is quite beneficial. Ultimately combining multiple techniques such as post-processing pre-trained embeddings with the modified VecMap algorithm is proven to be expedient to obtain superior results in BLI.

In the future, we plan to further improve the model to accommodate morphology-rich, high inflectional natures of LRLs similar to Sinhala. Furthermore, we are intending to incorporate the results obtained by this research into a task-specific downstream task such as Neural Machine Translation, Named Entity Recognition, or sentiment analysis to improve the results of the entire task.

REFERENCES

[1] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[2] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.

[3] Anushika Liyanage, Surangika Ranathunga, and Sanath Jayasena. Bilingual lexical induction for sinhala-english using cross lingual embedding spaces. In *2021 Moratuwa Engineering Research Conference (MERCon)*, pages 579–584. IEEE, 2021.

[4] Wikipedia contributors. Bilingual lexicon — Wikipedia, the free encyclopedia, 2019. [Online; accessed 22-March-2022].

[5] Nan Jiang. Lexical representation and development in a second language. *Applied linguistics*, 21(1):47–77, 2000.

[6] Davide Turcato. Automatically creating bilingual lexicons for machine translation from bilingual text. *arXiv preprint cmp-lg/9807010*, 1998.

[7] Dhouha Bouamor, Nasredine Semmar, and Pierre Zweigenbaum. Automatic construction of a multiword expressions bilingual lexicon: A statistical machine translation evaluation perspective. In *Proceedings of the 3rd Workshop on Cognitive Aspects of the Lexicon*, pages 95–108, 2012.

[8] Ruotian Ma, Minlong Peng, Qi Zhang, and Xuanjing Huang. Simplify the usage of lexicon in chinese ner. *arXiv preprint arXiv:1908.05969*, 2019.

[9] Ann Irvine and Chris Callison-Burch. A comprehensive analysis of bilingual lexicon induction. *Computational Linguistics*, 43(2):273–310, 2017.

[10] Surangika Ranathunga, En-Shiun Annie Lee, Marjana Prifti Skenduli, Ravi Shekhar, Mehreen Alam, and Rishemjit Kaur. Neural machine translation for low-resource languages: A survey. *arXiv preprint arXiv:2106.15115*, 2021.

[11] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, 2017.

[12] Yoav Goldberg and Graeme Hirst. Neural network methods in natural language processing. morgan & claypool publishers(2017). *9781627052986 (zitiert auf Seite 69).*

[13] Sahar Ghannay, Benoit Favre, Yannick Esteve, and Nathalie Camelin. Word embedding evaluation and combination. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 300–305, 2016.

[14] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[16] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[17] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[18] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, 2015.

[19] Peter H Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[20] Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*, 2017.

[21] Mikel Artetxe, Gorka Labaka, Inigo Lopez-Gazpio, and Eneko Agirre. Uncovering divergent linguistic information in word embeddings with lessons for intrinsic and extrinsic evaluation. *arXiv preprint arXiv:1809.02094*, 2018.

[22] Ivan Vulić, Anna Korhonen, and Goran Glavaš. Improving bilingual lexicon induction with unsupervised post-processing of monolingual word vector spaces. In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 45–54, 2020.

[23] Vikas Raunak, Vivek Gupta, and Florian Metze. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, 2019.

[24] Christopher M Biship. Pattern recognition and machine learning (information science and statistics), 2007.

[25] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

[26] Enrique Alfonseca, Massimiliano Ciaramita, and Keith Hall. Gazpacho and summer rash: lexical relationships from temporal patterns of web search

queries. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1046–1055, 2009.

[27] Taylor Berg-Kirkpatrick and Dan Klein. Simple effective decipherment via combinatorial optimization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 313–321, 2011.

[28] Alexandre Klementiev and Dan Roth. Weakly supervised named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 817–824, 2006.

[29] Charles Schafer and David Yarowsky. Inducing translation lexicons via diverse similarity measures and bridge languages. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[30] Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012*, pages 1459–1474, 2012.

[31] Stephan Gouws and Anders Søgaard. Simple task-specific bilingual word embeddings. In *HLT-NAACL*, pages 1386–1390, 2015.

[32] Omer Levy, Anders Søgaard, and Yoav Goldberg. A strong baseline for learning cross-lingual word embeddings from sentence alignments. *arXiv preprint arXiv:1608.05426*, 2016.

[33] Tianze Shi, Zhiyuan Liu, Yang Liu, and Maosong Sun. Learning cross-lingual word embeddings via matrix co-factorization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 567–572, 2015.

[34] Aitor Ormazabal, Mikel Artetxe, Gorka Labaka, Aitor Soroa, and Eneko Agirre. Analyzing the limitations of cross-lingual word embedding mappings. *arXiv preprint arXiv:1906.05407*, 2019.

[35] Liangchen Wei Zhi-Hong Deng. A variational autoencoding approach for inducing cross-lingual word embeddings. 2017.

[36] Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. Bilingual lexicon induction with semi-supervision in non-isometric embedding spaces. *arXiv preprint arXiv:1908.06625*, 2019.

[37] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[38] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

[39] Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970, 2017.

[40] Mladen Karan, Ivan Vulić, Anna Korhonen, and Goran Glavaš. Classification-based self-learning for weakly supervised bilingual lexicon induction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6915–6922, 2020.

[41] Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*, 2014.

[42] Nikola Mrkšić, Diarmuid O Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve

Young. Counter-fitting word vectors to linguistic constraints. *arXiv preprint arXiv:1603.00892*, 2016.

[43] Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. Neural-based noise filtering from word embeddings. *arXiv preprint arXiv:1610.01874*, 2016.

[44] Igor Labutov and Hod Lipson. Re-embedding words. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 489–493, 2013.

[45] Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. Ultradense word embeddings by orthogonal transformation. *arXiv preprint arXiv:1602.07572*, 2016.

[46] Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv preprint arXiv:1702.01417*, 2017.

[47] Takashi Wada, Tomoharu Iwata, Yuji Matsumoto, Timothy Baldwin, and Jey Han Lau. Learning contextualised cross-lingual word embeddings and alignments for extremely low-resource languages using parallel corpora. *arXiv preprint arXiv:2010.14649*, 2020.

[48] Goran Glavaš and Ivan Vulić. Non-linear instance-based cross-lingual mapping for non-isomorphic embedding spaces. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7548–7555, 2020.

[49] MD Rajitha, LL Piyarathna, MMD Nayanajith, and S Surangika. Sinhala and english document alignment using statistical machine translation. In *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 29–34. IEEE, 2020.

[50] Aloka Fernando, Surangika Ranathunga, and Gihan Dias. Data augmentation and terminology integration for domain-specific sinhala-english-tamil statistical machine translation. *arXiv preprint arXiv:2011.02821*, 2020.

[51] Yanyang Li, Yingfeng Luo, Ye Lin, Quan Du, Huizhen Wang, Shujian Huang, Tong Xiao, and Jingbo Zhu. A simple and effective approach to robust unsupervised bilingual dictionary induction. *arXiv preprint arXiv:2011.14874*, 2020.

[52] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*, 2014.

[53] Tasnim Mohiuddin, M Saiful Bari, and Shafiq Joty. Lnmap: Departures from isomorphic assumption in bilingual lexicon induction through non-linear mapping in latent space. *arXiv preprint arXiv:2004.13889*, 2020.

[54] Anders Søgaard, Sebastian Ruder, and Ivan Vulić. On the limitations of unsupervised bilingual dictionary induction. *arXiv preprint arXiv:1805.03620*, 2018.

[55] Long Duong, Hiroshi Kanayama, Tengfei Ma, Steven Bird, and Trevor Cohn. Learning crosslingual word embeddings without bilingual corpora. *arXiv preprint arXiv:1606.09403*, 2016.