Humboldt-Universität zu Berlin
Faculty of Mathematics and Natural Sciences
Computer Science Department

Bilingual Lexicon Induction for Oshiwambo

Bachelor's Thesis

A thesis presented for the degree of
Bachelor of Science (B. Sc.)

submitted by: Adrian Johannes Breiding
born on: 09.08.2000
born in: Kiel

supervisors: Prof. Dr. Alan Akbik
Prof. Dr. Torsten Hiltmann

submitted on: ..... defended on: .....

Acknowledgments

I want to thank my Namibian friends, who donated some of their time to complete multiple questionnaires consisting of about 350 questions each and answered my vocabulary questions (once again). Without this effort, this thesis would not have been possible. Thank you, Tangeni Heelu, Kronelia Kamanya, Olivia N. Kamati, Eva Ndengu, Taimi M. Ndahutuka and Matheus P. Nekongo.

Abstract. Traditionally, in Natural Language Processing, parallel and sentence-aligned corpora are required for translation. Bilingual lexicon induction (BLI) induces translations based on monolingual data, optionally with additional seed translations. This approach poses fewer restrictions on the data, yielding a system with broader applicability. Various experiments confirm its potential, especially for related and/or high-resource languages. Hence, recent research focuses on the low-resource and linguistically distant setting.

In this thesis, we test an established approach combining a graph-based and a linear mapping view on BLI and its capability to perform BLI for two African languages of the Bantu language family: Swahili and Oshiwambo. We hypothesize that for agglutinative languages, it can be beneficial to leverage sub-word embeddings and further investigate the influence of corpus sizes and isomorphy metrics on the performance of BLI. However, we expect complications in BLI caused by the agglutinative nature of the languages, where composed words cannot be correctly split in a semantically correct manner.

The generated Oshiwambo dictionary scores 1.65, 4.35 and 6.86 in $P@1$, $P@5$ and $P@10$ respectively. We further compute a mean reciprocal rank of 2.97. These scores are lower than those of previous work on BLI for Oshiwambo. The dictionary for Swahili scores on average 39.18 in $P@1$, a competitive result demonstrating the potential for the system to be applied to BLI for Bantu languages. The results remain inconclusive on the possible advantage of using sub-word embeddings over word embeddings. Limited insight can be gained into the influence of the isomorphy metrics, but the results suggest a contribution to the lower scores in Oshiwambo. The hypothesis about complications due to the agglutinative nature of Bantu languages could not be verified.

List of Figures

List of Tables

Contents

1. Introduction

This introductory chapter presents the concept of Bilingual Lexicon Induction (BLI), the motivation behind its application to the Namibian language Oshiwambo, and a short introduction to the language itself. It further includes a brief presentation of the main contributions and an outline of this thesis.

1.1. Motivation

As of February 2025, Google Translate supports 249 languages (Google Ireland Limited). Even if Google Translate does not support a particular language, chances are other helpful online resources exist. But what options do speakers or learners of a language have who do not have any resources simply because they do not exist or are difficult to acquire? With more or less one language going extinct every month (Bromham et al., 2022), there should be an interest in documenting and building up resources for all languages, especially the endangered ones.

I was one of those almost resourceless students when I was trying to learn Oshikwanyama. Oshikwanyama is part of a collection of dialects named Oshiwambo, which are mainly used by the Ovambo peoples in northern Namibia and southern Angola and are attributed to the Bantu language family (Haugh, 2022). According to the 2011 census, 48.9 % of Namibian households (with a total population of 2,113,077) speak an Oshiwambo language (Namibia Statistics Agency (NSA), 2013). Two of the dialects belonging to the Oshiwambo family, Oshikwanyama and Oshindonga, have been developed as standard languages and are actively used in education and culture (Haugh, 2022).

(a) (b)

Figure 1: Distribution of Oshindonga (a) and Oshikwanyama (b) speakers in Namibia (Atlas of Namibia Team, 2022)

With this institutional standing, the languages are not considered endangered (Eberhard et al., 2024). Yet, to my knowledge, there are no online dictionaries or translation tools,

and only a couple of physical, partially outdated dictionaries (refer to section 2) available. Hence, the goal of this thesis is to use the limited amount of resources available to create an Oshiwambo-English dictionary with the means of *bilingual lexicon induction* (BLI).

Typically, in Natural Language Processing, translations are learned from parallel, sentence-aligned corpora. BLI instead attempts to induce the translations from potentially unrelated monolingual corpora, usually with the aid of a small seed dictionary, significantly reducing the data requirements (Irvine and Callison-Burch, 2017). While there are various approaches, most techniques, including the one I will use in this thesis, rely on separately generating vector representations (embeddings) of the words in the training data for each language and then trying to find a mapping between these embeddings.

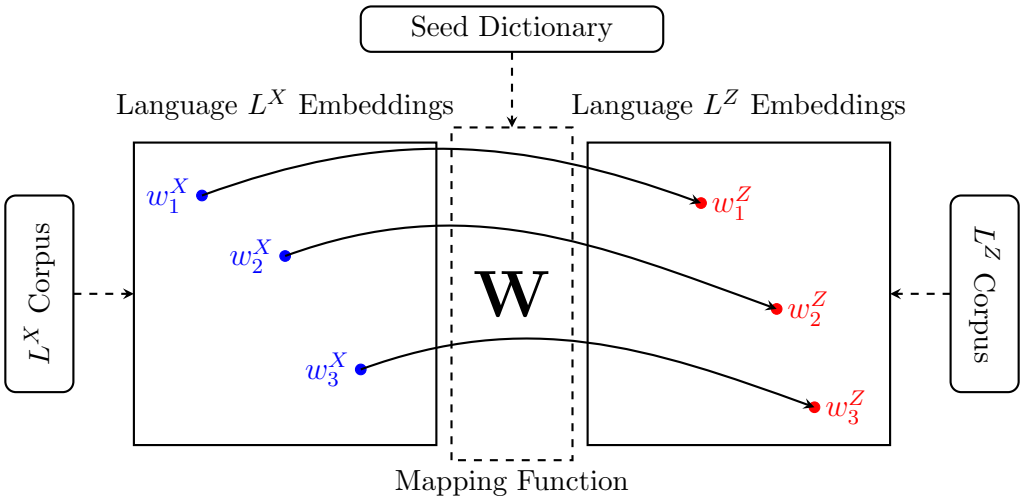

Figure 2: Schematic visualization of bilingual lexicon induction using embeddings [GPT]

A popular strategy to compute this mapping W is solving the orthogonal Procrustes problem, which tries to find a linear map from the source to the target embedding space. Throughout this thesis, we will refer to this as the Procrustes approach. This works well enough for roughly isomorphic embedding spaces but worsens as the spaces diverge. Note that isomorphic in this context is understood as "geometrically similar". The assumption that such a W exists is commonly called the *isomorphism hypothesis* (Søgaard et al., 2018). Alternatively, we can derive weighted graphs based on the similarity of the embeddings (Marchisio et al., 2021). In this case, we try to solve the graph-matching problem, i.e., find a mapping between the nodes of the graphs that minimizes the edge disagreements. We denote this technique as the graph-based approach.

1.2. Problem Statement

To succeed with BLI for Oshiwambo, two main challenges must be addressed. First, one has to find suitable data. Oshiwambo is not necessarily considered to be low-resourced, thanks to translation efforts, e.g., yielding educational material, but the resources that exist are often not easily accessible and/or difficult to parse, effectively rendering the language low-resourced nevertheless (Nekoto et al., 2022). Typical multilingual data sources, such as governmental websites, are, at least for now, only available in English, the only official language in Namibia (Ministry of Home Affairs, Immigration, Safety & Security). Joshi et al. (2020) ambiguously classify Oshindonga, as a "left-behind" and "scraping-by" language in what they call the "language resource race".

The second main issue is the linguistic distance between Oshiwambo and English. Oshiwambo, being a part of the Bantu language family (Haugh, 2022), is an agglutinative language (Wald, 2003). To understand the implications, consider the word *Aame*, which is Oshikwanyama and translates to *It's me*. Adding the prefix *Ha-* negates the expression yielding ***Ha****ame* which corresponds to *It's* ***not*** *me* (Crane et al., 2004). This trait suggests that the isomorphism hypothesis does not necessarily need to hold in the case of Oshiwambo.

1.3. Main Contributions

Generalizing from the concrete example of Oshiwambo, this thesis attempts BLI in a low-resource and linguistically distant setting. In addition to performing experiments on the language pair Oshiwambo-English, we also investigate Oshiwambos' better-resourced linguistic relative, Swahili, to have a higher resource language as a reference. Using the `Fundus` (Dallabetta et al., 2024) library, we create primarily newspaper-based monolingual corpora for the two language pairs. We generate seed dictionaries from scanned print dictionaries in the case of Oshiwambo and fall back to `Google Translate` for Swahili.

We suggest using embeddings trained on a sub-word level to better represent the agglutinative nature of the Bantu language family and further use them to test the performance of a combination of the graph-based and Procrustes approaches (Marchisio et al., 2022). This system combination has already been tested on (mostly) Eurasian languages and has proven successful in the low-resource setting and for distant language pairs. This thesis now evaluates it for two African languages and generates two 3000 word dictionaries.

We further assume complications in BLI related to Oshiwambo's agglutinative nature and test that hypothesis with a dedicated question type in the manual evaluation. Lastly, we analyze the performance of our approach to BLI with respect to two different metrics measuring the degree of isomorphy of the two embedding spaces.

1.4. Outline

The thesis is comprised of 4 different parts:

Chapter 2: Related Work. This chapter starts by providing an overview of different systems for training and refining embeddings. Based on those embeddings, two perspectives on bilingual lexicon induction and their corresponding solutions are provided. Next, the isomorphism hypothesis is discussed in a more formal setting. The chapter ends with presenting possible data sources and two other publications that have addressed Oshiwambo in the context of BLI and machine translation. A strong focus lies in providing the theoretical framework behind the most important concepts.

Chapter 3: Methodology. This section provides an overview of the steps for BLI for Oshiwambo-English. The data acquisition and the necessary preprocessing steps are discussed in the first step, followed by an introduction to the chosen tokenization algorithm. The chapter ends with an explanation of one iteration in the bilingual lexicon induction algorithm.

Chapter 4: Evaluation. First, the evaluation methodology is illustrated. Next, the individual experiments are presented and explained with regard to the main goals of this thesis, and finally, the overall findings are presented.

Chapter 5: Discussion The thesis concludes with a summary and discussion of the results and ends with an outlook on possible future research questions.

2. Literature Review

This chapter introduces selected basic embedding training mechanisms, the underlying theoretical formalism, and strategies for embedding refinement. It then provides a similar overview of different approaches to BLI, emphasizing the mathematical background relied on. Next, the isomorphism hypothesis is formally introduced. The chapter ends by presenting potential data sources and other relevant work on the Oshiwambo languages.

2.1. Bilingual Lexicon Induction

One of the elementary questions in natural language processing is how natural language can be represented to preserve its intricacies and information density while allowing efficient processing with computational methods.

2.1.1. Embedding Techniques

The established choice for text is real-valued vectors. The simplest form is a *one-hot vector* (Lavrač et al., 2021). In this encoding, a word $w_j, j \in \{0, \cdots, n-1\}$ is represented as a vector $x \in \mathbb{R}^n$, where n is the number of words in the vocabulary. All entries x_i are set to zero, except for x_j, which is set to one. It is easy to see that there are some issues with this representation. Semantic relationships, for example, are not captured in this encoding - all vectors have the same Euclidean distance from each other (Lavrač et al., 2021).

Current methods to generate word embeddings better at preserving semantic correlations rely on the distributional hypothesis: Harris (1954) theorizes that words in similar contexts often have similar meanings. Mikolov et al. (2013b) presented one method incorporating this hypothesis, where a single-layer feed-forward neural network is used to predict a word given its context. The method is called `Word2Vec` and has two variants: Continous Bag of Words (`CBOW`), where the central word of a text sequence is predicted, and `skip-gram`, where the neighborhood is predicted using the word. Figure 3 provides a schematic visualization of the latter model.

More precisely, for a sequence of training words $w_1, w_2, \cdots, w_T$, the `skip-gram` model tries to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{1}$$

where the probability of predicting w_O given w_i is defined as:

$$p(w_O|w_I) = \frac{\exp u_{w_O}^{\top} v_{w_I}}{\sum_{w=1}^{W}\exp u_w^{\top} v_{w_I}} \tag{2}$$

u_w and v_w are the "output" and "input" vector representations of w, while W is the

total number of words (Mikolov et al., 2013b). Interestingly, embeddings generated using the `skip-gram` model show a degree of language understanding, which can be observed using basic mathematical operations. In their experiments, Mikolov et al. (2013b) found, for example, that computing "Madrid" - "Spain" + "France" and extracting the nearest neighbor yields "Paris".

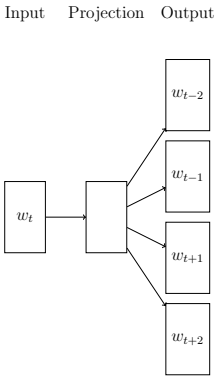

Figure 3: The `skip-gram` model architecture. Adapted from Mikolov et al. (2013b) [GPT]

A different, older approach, Latent Semantic Analysis (LSA) (Deerwester et al., 1990), uses a *term-document matrix* as a foundation. The entries of this matrix $(X_{ij}) \in \mathbb{N}^{W \times D}$, where W is the size of the vocabulary and D is the number of documents in the training data, contain the frequency of the word w_i in the document d_j. This matrix will, in general, be sparse, so LSA compresses the information using the compact Singular Value Decomposition in the following steps (Lavrač et al., 2021):

1. Factorization of the term-document matrix: $X = W\Sigma C$ into a semi-orthogonal word matrix $W \in \mathbb{R}^{W \times m}$, a semi-orthogonal context matrix $C \in \mathbb{R}^{m \times D}$ and the diagonal matrix $\Sigma \in \mathbb{R}^{m \times m}$, containing the singular values.

2. The $m - d$ lowest singular values are set to 0, effectively truncating W, Σ and C. The truncated $\tilde{W} \in \mathbb{R}^{W \times d}, \tilde{\Sigma} \in \mathbb{R}^{d \times d}, \tilde{C} \in \mathbb{R}^{d \times D}$ approximate $X \approx \tilde{X} = \tilde{W}\tilde{\Sigma}\tilde{C}$.

3. The embedding of the word w_i corresponds to the i-th row of $\tilde{W}\tilde{\Sigma}$ and has dimension d (Deerwester et al., 1990).

The comparison of the cooccurrence of two terms can then be represented as the dot product of the two corresponding row vectors of $\tilde{W}\tilde{\Sigma}$, or in general by computing

$$\tilde{X}\tilde{X}^\top = \tilde{W}\tilde{\Sigma}^2\tilde{W}^\top \tag{3}$$

and observing the entry (i, j).

Both of the presented methods have their limits: by their very nature, LSA captures the global cooccurrence very well but performs poorly on word analogy tasks. Meanwhile,

`skip-gram` performs stronger on those tasks but cannot reflect the statistics of the entire corpus, as the context size limits it.

Pennington et al. (2014) propose the `GloVe` model, which stands for *Global Vectors* and combines the two underlying ideas. This approach is based on the *term-term matrix* X, whose entries X_{ij} represent the number of times the word w_i occurs in the context of word w_j and the cooccurrence probabilities $P_{ij} := P(j|i) = \frac{X_{ij}}{X_i}$, where $X_i = \sum_k X_{ik}$. The authors' intuitive approach considers two words w_i, w_j, probed with another word $\tilde{w}_k$. Say $\tilde{w}_k$ were closely related to w_i, but not w_j, one expects $\frac{P_{ik}}{P_{jk}} \gg 1$, and similarly with switched roles. Also, if $\tilde{w}_k$ were related or distant to both w_i and w_j, we would expect $\frac{P_{ik}}{P_{jk}} \approx 1$. We also want our model $F(w_i, w_j, \tilde{w}_k)$ to preserve the simple, linear structure of vector spaces, hence place the following restrictions on it:

$$F\left((w_i - w_j)^\top \tilde{w}_k\right) = \frac{P_{ik}}{P_{jk}} \tag{4}$$

Next, notice that the distinction between a context word $\tilde{w}_k$ and a word w_i is somewhat arbitrary, and the two roles can be exchanged by applying $w \leftrightarrow \tilde{w}$. Note that, to be consistent, we also need to apply $X \leftrightarrow X^\top$. To have a model that is invariant under this operation, we first require that $F \in \mathrm{Hom}((\mathbb{R}, +), (\mathbb{R}_{>0}, \times))$ is a group homomorphism:

$$F\left((w_i - w_j)^\top \tilde{w}_k\right) = \frac{F(w_i^\top \tilde{w}_k)}{F(w_j^\top \tilde{w}_k)} \tag{5}$$

Using equation 4, we get for F:

$$F(w_i^\top \tilde{w}_k) = P_{ik} = \frac{X_{ik}}{X_i} \tag{6}$$

Also, equation 5 is solved with the exponential function, implying

$$w_i^\top \tilde{w}_k = \ln(P_{ik}) = \ln(X_{ik}) - \ln(X_i) \tag{7}$$

As this is not yet symmetric, we define a bias b_i absorbing $\ln(X_i)$ term and add a bias term $\tilde{b}_k$ for $\tilde{w}_k$, yielding:

$$w_i^\top \tilde{w}_k + b_i + \tilde{b}_k = \ln(X_{ik}) \tag{8}$$

To address the issue, that $\ln(x)$ is not defined for $x = 0$, the `GloVe` cost function is given as follows:

$$J = \sum_{i=1}^{V} \sum_{j=1}^{V} f\left(X_{ij}\right) \left(w_i^\top \tilde{w}_j + b_i + \tilde{b}_j - \ln(X_{ij})\right)^2 \tag{9}$$

where $f(\cdot)$ obeys the following properties:

- $f(0) = 0$
- If f is viewed as a continuous function, it should hold that $\lim_{x \to 0} f(x) \ln(x)^2 < \infty$
- $f(x)$ is not decreasing to not overweight rare cooccurrences
- $f(x)$ is comparatively small for large x to not overweight frequent cooccurrences

Experiments by Pennington et al. (2014) suggest the choice of

$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}}\right)^{\alpha} & , \text{if} x < x_{\max} \\ 1 & , \text{else} \end{cases} \tag{10}$$

where $x_{\max} = 100$ and $\alpha = \frac{3}{4}$.

2.1.2. Embedding Refinement

Embeddings are essential for all approaches to BLI and are a critical factor in determining the algorithm's success. Apart from the somewhat controversial isomorphism hypothesis, another issue emerges with higher dimensional embeddings, known as the *hubness* problem. This means that specific points are exceptionally often among the k nearest neighbors in data sets of higher dimensionality. These points are called *hubs* and are problematic since they may push down correct translations in the nearest neighbor list of a nearby word (Radovanović et al., 2010; Dinu and Baroni, 2014). Figure 4 demonstrates this behavior on three real datasets. Observe the increasing skewness of the probability distribution for a data point being located within the k nearest neighbors of N_k other points: In high dimensional data sets, an arbitrary data point is substantially more likely to only be within the list of k nearest neighbors for only a few other data points.

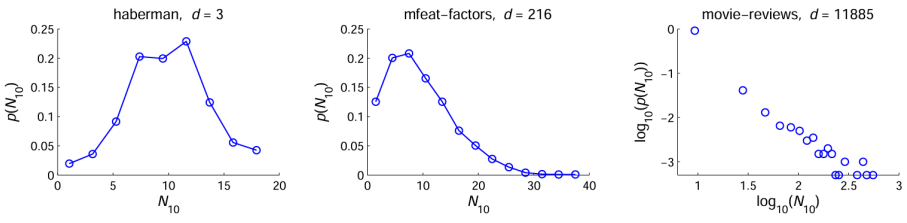

Figure 4: Empirical distribution of the number of occurrences of a data point in the top $k = 10$ nearest neighbors N_k for three real data sets of varying dimensionality. Reproduced under CC-BY license from Radovanović et al. (2010)

For the embeddings to be less sensitive to hubness and potentially more isomorphic, training them on a parallel, word-aligned corpus is possible. These new cross-lingual word embeddings render mappings obsolete. Experiments show higher $P@1$ scores in BLI

for the four tested language pairs compared to the `VecMap` approach (Ormazabal et al., 2019). While this proves to be a possible solution for languages with suitable corpora, this will likely not help in low-resource contexts.

Alternatively, for a low-resource language L^X and a linguistically distant language L^Z, it can be beneficial to consider a higher-resource language L^Y related to L^X and, without supervision, compute a transformation matrix between their embedding spaces X and Y generating an embedding space $\tilde{X}$ aligned with Y (Sannigrahi and Read, 2022). In the following step, the high-resource nature of languages L^Y and L^Z can be exploited by applying a joint-training approach yielding an embedding space $\tilde{Z}$, also aligned with Y. In the final step, it can reasonably assumed that $\tilde{X}$ and $\tilde{Z}$ are isomorphic and a transformation matrix mapping $\tilde{X}$ to $\tilde{Z}$ is determined.

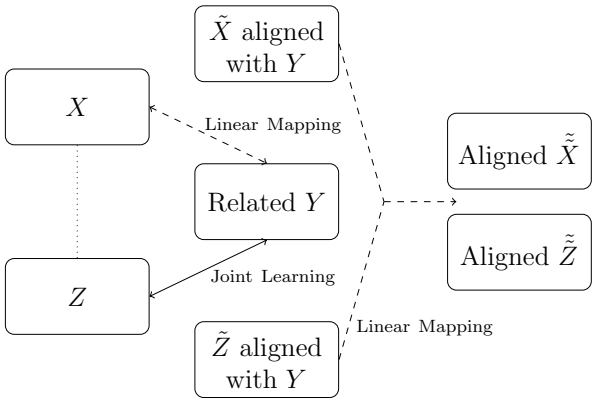

Figure 5: Visualization of embedding space alignment using a higher-resource related language, where dashed lines symbolize lacking parallel data. Adapted from Sannigrahi and Read (2022) [GPT]

Ding et al. (2024) present another approach to enhance the isomorphism between the two embedding spaces. They propose generating "pseudo-sentences" where synonyms replace certain words from existing sentences in the corpus. These synonyms are grouped together as related tuples and stored in a monolingual synonym lexicon $\mathcal{D}^X_{\text{syn}} = \{(w^x_{k_1}, w^x_{k_2}, w^x_{k_3}), \cdots, (w^x_{k_{n-2}}, w^x_{k_{n-1}}, w^x_{k_n})\}$. The original sentences s_i are extracted from the corpus, and if a word $w_i \in \mathcal{D}_{\text{syn}}$ has synonyms $S_i = (w_1, \cdots, w_N)$, the k words in S_i occurring most frequently in the corpus are chosen to replace w_i in the sentence s_i, creating k pseudo-sentences. The embeddings trained on the original corpus extended with pseudo-sentences are closer for synonymous words, as they regularly appear in similar contexts. The application of the method reduces the Hausdorff distance, a measure for the isomorphy of two (metric) vector spaces.

As already mentioned, Oshiwambo is an agglutinating language. Intuitively, it makes sense to try and generate embeddings by considering the syllables in one way or another. Experiments with Korean and Swahili, both (highly) agglutinative languages, confirm

this and show a higher performance when using syllable-aware embeddings (Shikali et al., 2019; Choi et al., 2017). In both settings, the researchers generated trained syllable vectors in combination with a convolutional neural network. More concretely, in the case of Swahili, every word is broken down into elements of the *Swahili Syllabic Alphabet* S, which are represented as one-hot vectors and projected into the embedding space yielding a composed word embedding $E_w = [E_{s_1}, \cdots, E_{s_l}]$ from the syllable embeddings $E_{s_i} \in \mathbb{R}^{d \times 1}$ (Shikali et al., 2019). E_w is further considered a matrix and a convolution of width n is applied between it and $F \in \mathbb{R}^{d \times n}$ generating a feature map $f^k \in \mathbb{R}^{l-n+1}$. More precisely, with a bias b:

$$f_i^w = \tanh\left(\mathrm{tr}\left((E_w)_{*,i:i+n-1} \cdot F^T\right) + b\right) \tag{11}$$

For filters of varying width, the final word embeddings are then derived as $w_t = [y_1^w, \cdots, y_h^w]$, where $y^w = \max_i f_i^w$ and h is the number of applied filters. A following highway network processes interactions between the sequence of syllables. Given these *syllable aware* word embeddings, a long short-term memory (LSTM) language model is trained to predict the next word of a given context accurately. In this task, the syllable-aware embeddings receive better perplexity scores than character-based embeddings. They also perform better on a word analogy task, suggesting they potentially improve performance in further NLP contexts.

Another alternative is Byte-Pair embeddings, which have the advantage of requiring a lower amount of resources compared to other subword unit embedding approaches (Sennrich et al., 2016; Heinzerling and Strube, 2018). It is based on Byte-Pair encoding, where a new symbol iteratively replaces the most common symbol pair. After encoding, the symbols should represent the most common character strings. Training embeddings on these symbols will result in them reflecting the most frequent subwords in the hope of corresponding to semantically significant morphemes. By varying the number of merge operations o, the resulting embeddings can be manipulated to represent shorter character sequences or many frequently occurring words primarily.

2.1.3. The Procrustes Approach

Mikolov et al. (2013a) presented the original idea that later led to the Procrustus method after the discovery that embeddings can capture linguistic regularities, e.g., computing "king" - "man" + "woman" with the respective vector representations results in a vector close to "queen". Hence, postulating a mapping from the source embedding space to the target embedding space seemed reasonable.

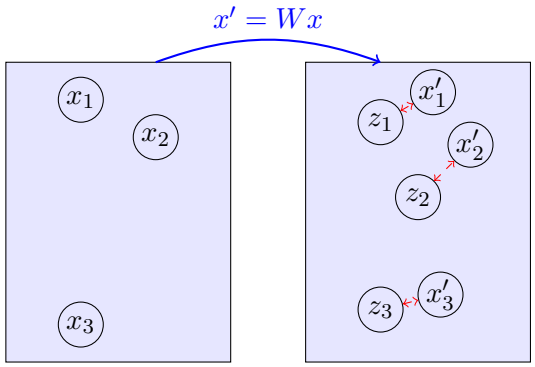

Figure 6: Visualization of the Procrustes approach. Adapted from Marchisio et al. (2021) [GPT]

Concretely, given a set of word pairs and their corresponding embeddings $\{x_i, z_i\}_{i=0}^{n-1}$ with $x_i, z_i \in \mathbb{R}^d$, the goal is to find a matrix $W \in \mathbb{R}^{d \times d}$, such that Wx_i is a good approximation of z_i. This can be represented as the following optimization problem:

$$\arg\min_{W \in \mathbb{R}^{d \times d}} \sum_{i=0}^{n-1} \|Wx_i - z_i\|_2^2 \tag{12}$$

with $\|\cdot\|_2 = \sqrt{\langle \cdot, \cdot \rangle_2}$. This problem can then be solved using a gradient descent approach. Given such W, a new word and its vector representation x, we can then compute $Wx = z$ and determine possible translation candidates using a nearest neighbor search by using cosine similarity, which is defined as follows for $x, z \in \mathbb{R}^d$

$$s_{\cos}(x, z) = \frac{\langle x, y \rangle_2}{\|x\|_2 \, \|y\|_2} \tag{13}$$

and can be formulated as a metric by

$$d_{\cos}(x, z) = 1 - s_{\cos}(x, z) \tag{14}$$

Despite its simplicity, this method is quite effective (Mikolov et al., 2013a). However, attentive readers may have noticed a slight disparity when comparing equations 12 and 13: the measure of distances in training fundamentally differs from the measure used in extraction. Xing et al. (2015) suggest fixing this by training normalized embeddings and redefining equation 12 as:

$$\arg\max_{W \in O(d)} \sum_{i=1}^{n} (Wx_i)^\top z_i \tag{15}$$

for normalized x_i, z_i. If we further require $W \in O(d)$ to be an orthogonal matrix, this normalization ensures the equivalence to the cosine distance. Experiments showed that

this constraint leads to gains in the $P@1$ and $P@5$ scores in BLI compared to the unnormalized approach.

By reformulating equation 12 using the Frobenius norm and incorporating the previous insights, we receive

$$\underset{W \in O(d)}{\arg\min} \|WX - Z\|_F^2 \tag{16}$$

which is equivalent to the Procrustes Problem.

Definition 1 (Orthogonal Procrustes Problem (Schönemann, 1966))**.** *For two matrices $X, Z \in \mathbb{R}^{m \times n}$, the Orthogonal Procrustes Problem is defined as:*

$$\underset{W \in O(n)}{\arg\min} \|XW - Z\|_F^2$$

The Orthogonal Procrustes Problem can be solved exactly using the Singular Value Decomposition (SVD) (Conneau et al., 2018):

$$W = \underset{W \in O(d)}{\arg\min} \|WX - Z\|_F^2 = UV^\top \qquad UD^{\frac{1}{2}}V^\top = \text{SVD}(ZX^\top) \tag{17}$$

The detailed derivation can be found in section A.

Having found a mapping between the embedding spaces, we can now use it to predict translations. In the original paper, this was done by extracting the nearest neighbors using the cosine similarity metric. Using nearest neighbors can raise a minor paradox since it is possible for a vector x to be within the k nearest neighbors of y, while the converse does not hold. Especially in higher dimensional spaces, it can be observed that so-called *hubs* and *anti-hubs* are formed. These hubs are the nearest neighbors for many other points with a high probability, whereas anti-hubs are not the nearest neighbors for any other point (Radovanović et al., 2010). Naturally, this causes issues when extracting potential translation candidates using a standard nearest-neighbor measure.

To produce a more reliable matching, Conneau et al. (2018) suggest to consider a bi-partite graph, i.e., a graph[1] $(V, E \subset V \times V)$, whose nodes V can be split into two disjoint sets $U_1 \sqcup U_2 = V$, such that for all edges $(x, y) \in E$ it holds that $x \in U_1 \wedge y \in U_2$ or $x \in U_2 \wedge y \in U_1$. In this graph, each word in one language is connected with the k nearest neighbors of the other language. They further define $\mathcal{N}_T(Wx_s)$ the neighborhood associated with the mapped source word embedding Wx_s. By definition, this neighborhood only contains words in the target language. Analogously, you can define $\mathcal{N}_S(z_t)$ the neighborhood for words z_t in the target language.

[1] A graph is formally introduced in section 2.3.

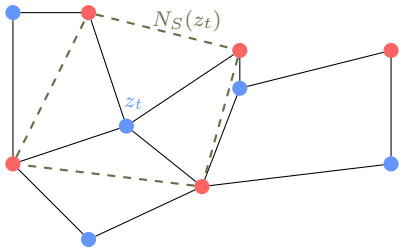

Figure 7: Visualization of the neighborhood $N_S(z_t)$ for a word z_t in the target language for $k = 2$. [GPT]

We can now denote:

$$r_T(Wx_s) = \frac{1}{K} \sum_{z_t \in \mathcal{N}_T(Wx_s)} s_{\cos}(Wx_s, z_t) \qquad r_T(z_t) = \frac{1}{K} \sum_{Wx_s \in \mathcal{N}_S(z_t)} s_{\cos}(Wx_s, z_t) \quad (18)$$

as the mean similarity of a source (target) embedding to its target (source) neighborhood. Using this, we can now define the Cross-Domain Similarity Local Scaling Measure (CSLS):

Definition 2 (Cross-Domain Similarity Local Scaling Measure (Conneau et al., 2018))**.** *Given the mean similarities for an embedding to its neighborhood $r_T(Wx_s), r_S(z_t)$, the Cross-Domain Similarity Local Scaling Measure is defined as*

$$\text{CSLS}(Wx_x, z_t) = 2s_{\cos}(Wx_s, z_t) - r_T(Wx_s) - r_S(z_t) \quad (19)$$

Intuitively, this measure combats hubness by penalizing points with dense neighborhoods while boosting the similarity values for anti-hubs. This intuition is also confirmed by experiments in word translation retrieval (Conneau et al., 2018).

The underlying assumption is that the embedding spaces are "isomorphic", what is understood to be "geometrically similar" in this context. While it is reasonable for related languages, experiments have shown that it is less sensible in cases where the language similarity decreases or in cases of varying domains of the monolingual training data sets (Søgaard et al., 2018; Patra et al., 2019). To counteract the decreasing *degree of isomorphy*, there are options to make the embedding spaces "more isomorphic", some of which are presented in section 2.1.2.

Furthermore, it is possible to introduce and alter pre- and postprocessing and attempt different mapping algorithms to improve the obtained results. According to Li et al. (2023), one of the most representative BLI approaches based on cross-lingual word embeddings, which is especially effective in semi- and unsupervised settings, is `VecMap` (Artetxe et al., 2018). They normalize their embeddings before estimating an initial mapping by finding alternative representations $\tilde{X}, \tilde{Z} \in \mathbb{R}^{n \times m}$ of the word embedding matrices $X, Z \in \mathbb{R}^{n \times m}$, that are aligned across one axis. $\tilde{X}, \tilde{Z}$ and equations 20 and 21 are used to build an initial dictionary $D \in \{0, 1\}^{n \times n}$, with $D_{ij} = 1$ if the j-th word in

the target language corresponds to the i-th word source language. Next, two steps are iterated until convergence. The first step is to compute

$$\arg\max_{W_X, W_Z} \sum_{i,j} D_{ij}((X_{i*}W_X) \cdot (Z_{j*}W_Z)) \tag{20}$$

starting with the original embeddings. In the second step, the optimal dictionary over the similarity matrix of the mapped embeddings $XW_XW_Z^TZ^T$ is computed by

$$D_{ij} = \begin{cases} 1, & j = \arg\max_k (X_{i*}W_X) \cdot (Z_{k*}W_Z) \\ 0, & \text{else} \end{cases} \tag{21}$$

Some additional measures are necessary to avoid poor local minima. Finally, reweighting the embeddings in the target and source language further improves the results. In their experiments with four languages, they gain $6.72 - 13.20$ percentage points (p.p.) in accuracy compared to Mikolov et al. (2013a).

Once suitable mappings are available, they can be further exploited by more complex methods. In later work by Artetxe et al. (2019), they use a set of cross-lingual word embeddings and their corresponding monolingual corpora and suggest taking the n most frequent bigrams and trigrams of each language and assigning them the centroid of their components. These cross-lingual phrase embeddings are then used to generate a phrase table, which, combined with a distortion model and a 5-gram language model, results in a phrase-based machine translation system. After tuning, the model is used to translate the source language monolingual corpus, yielding a parallel corpus that can be word-aligned. As the final step, another phrase table is generated from the new corpus, and all non-unigram entries are discarded. In their experiments with Spanish, French, German, and Russian to English lexicon induction and $n = 400{,}000$, their method increases the $P@1$ score by around $5 - 8$ p.p. compared to the nearest neighbor method.

2.1.4. The Graph-Based Approach

To attempt BLI using graph matching, one can consider words as nodes in monolingual, weighted, undirected graphs[2] $G_X = (V_X, E_X, w_X), G_Z = (V_Z, E_Z, w_Z)$. The edge weights are computed using the cosine similarity metric. The fundamental idea is that relationships between words are reflected in the similarity of the corresponding embeddings, and these similarities do not vary much across different languages. With this assumption, one can try to find the optimal permutation π aligning $\pi(V_Z)$ with V_X. In the ideal case, we can find a π such that $\forall i, j \in E_X : (i, j) \in E_X \iff (\pi(i), \pi(j)) \in E_Z \wedge w_X(i, j) = w_Z(\pi(i), \pi(j))$. In a realistic setting, graphs are not isomorphic (for isomorphy as defined without weights); if they were, the weights would likely not align. Instead, one tries to minimize the edge disagreement.

[2] A graph, its adjacency matrix and the notion of isomorphy are formally introduced in section 2.3.

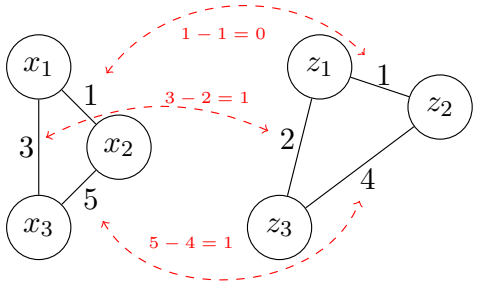

Figure 8: Visualization of the graph-based approach. Adapted from Marchisio et al. (2021) [GPT]

To quantify the edge disagreement of the two graphs, it is intuitive to choose a metric similar to:

$$\left\|A_X - PA_ZP^T\right\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}[(A_X)_{ij} - (PA_ZP^T)_{ij}]^2} \tag{22}$$

also known as the Frobenius norm.

Definition 3 (Graph Matching Problem (Marchisio et al., 2022))**.** *For two graphs G_X, G_Z and their corresponding adjacency matrices A_X, A_Z the problem*

$$\arg\min_{P\in\Pi_n} \left\|A_X - PA_ZP^T\right\|_F^2 \tag{23}$$

is called the Graph Matching Problem (GMP).

The problem is NP-hard, i.e., no known efficient algorithms exist, but functional approximations are available. The performance of this approach and the translation matrix method increases with the number of translation pairs, so-called seeds, that are passed into the algorithm. Experiments with various languages and varying numbers of seeds show that graph matching outperforms the nearest-neighbor method in most cases. The performance is further robust on dissimilar languages and low supervision (Marchisio et al., 2022).

2.1.5. Non-Embedding-Based Approach

With the rise of LLMs, there has also been research into using multilingual LLMs (mLLMs) for dictionary induction. Li et al. (2023) studied five families of mLLMs by building a dictionary through prompts like: *The word 'w^x' in L^y is: <mask>.* (a zero-shot prompt) or *The word w_1^x in L^y is w_1^y. The word w_2^x in L^y is w_2^y. The word w^x in L^y is <mask>.*

(a few-shot prompt) and comparing the results to approaches based on cross-lingual word embeddings. They show that their proposed few-shot prompting consistently receives higher $P@1$ scores. The major drawback is the limited scalability due to state-of-the-art mLLMs not being available for most languages for the moment.

2.2. Solving the Graph-Matching Problem

When solving the Procrustes Problem, we found a closed-form solution that was simple to compute. Unfortunately, there is no such solution for the GMP. However, we can use approximation algorithms to solve related problems, allowing us to find an approximate solution to the GMP. The following subsection is dedicated to iteratively reducing the GMP into Quadratic and Linear Assignment Problems and discussing the Hungarian (Kuhn, 1955) and FAQ (Vogelstein et al., 2015) algorithms for solving them. In the second step, the FAQ algorithm is extended to the GOAT (Saad-Eldin et al., 2021) and Seeded Graph Matching (Fishkind et al., 2019) algorithms, providing significantly better results in BLI.

2.2.1. Linear Assignment Problem

We will see that the GMP can essentially be broken down into some assignment problems, where the simple assignment problem can be illustrated as follows (Kuhn, 1955): Four workers $w \in \{1, 2, 3, 4\}$ need to split up four jobs $j \in \{1, 2, 3, 4\}$. Their qualifications can be represented by a qualification matrix Q:

$$Q = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where $Q_{wj} = 1$ if and only if the worker w is qualified for the job j. The problem now asks how many jobs can be assigned to qualified workers, with each worker doing a maximum of one job.

For n workers, the problem can be generalized as the General Assignment Problem (Kuhn, 1955) or as it is more commonly known (Linear) Assignment Problem (LAP).

Definition 4 (Linear Assignment Problem (Kuhn, 1955))**.** *For two sets $W = \{0, \cdots, n-1\}$, $J = \{0, \cdots, n-1\}$ and a rating matrix $R \in \mathbb{N}^{n \times n}$, let $\mathfrak{S}_n = \{\pi : W \to J \mid \pi \text{ is bijective}\}$ be the set of bijections. The Assignment Problem asks:*

$$\arg\max_{\pi \in \mathfrak{S}_n} \sum_{i=0}^{n-1} R_{i,m(i)} \tag{24}$$

Such a mapping π is called a permutation and can be expressed as a permutation

matrix P. Hence, one can appreciate the equivalent formulation of equation 24 as:

$$\arg\max_{P\in\Pi_n} \operatorname{tr}(PR) \tag{25}$$

where $\Pi_n = \{p_{ij} \in [0,1]^{n\times n}, \forall j \in \{0,\cdots,n-1\} : \sum_i p_{ij} = \sum_i p_{ji} = 1\}$ is the set of permutation matrices.

To find a solution to the LAP, (Kuhn, 1955) considers the dual problem, which can be stated as finding non-negative integers $u_0,\cdots,u_{n-1}$, $v_0,\cdots,v_{n-1}$ under the condition:

$$\forall i,j \in \{0,\cdots,n-1\} : u_i + v_j \geq R_{ij} \tag{26}$$

And computing:

$$\min_{u_i,v_j} \sum_{i=0}^{n-1} u_i + \sum_{j=0}^{n-1} v_j \tag{27}$$

2.2.2. The Hungarian Algorithm

Largely based on the works of D. König and E. Egerváry, two Hungarian mathematicians, the first algorithm we need is called *The Hungarian Algorithm* (Kuhn, 1955) and consists of the following steps:

1. Let $a = \sum_{i=0}^{n-1} \max_j R_{ij}$ be the sum over maximal values for each row and similarly $b = \sum_{j=0}^{n-1} \max_i R_{ij}$. We define a cover $\{u_i, v_j\}_{i,j}$:

$$\forall i \in \{0,\cdots,n-1\} : u_i = \begin{cases} \max_j R_{ij} & , a \leq b \\ 0 & , a > b \end{cases} \tag{28}$$

$$\forall j \in \{0,\cdots,n-1\} : v_j = \begin{cases} 0 & , a \leq b \\ \max_i R_{ij} & , a > b \end{cases} \tag{29}$$

   and a matrix Q, such that

$$Q_{ij} = \begin{cases} 1 & , u_i + v_j = R_{ij} \\ 0 & , \text{else} \end{cases} \tag{30}$$

   Finally, the rows of Q are examined in order, and the case of $a \leq b$, for the smallest j s.t. $\exists i : Q_{ij} = 1$ and $\forall i : Q_{ij} \neq 1^*$, the entry is redefined as $Q_{ij} = 1^*$. The instructions are identical, switching rows and columns in the second case of $a > b$.

2. Search the columns of Q in order for a 1^* or a 1, if there is no 1^* in the column.

As soon as a 1 is found in (i_1, j_0), begin with constructing a sequence similar to:

$$((i_1, j_0), 1), ((i_1, j_1), 1^*), ((i_2, j_1), 1), \cdots$$

There are two cases:

a) A 1 is found and recorded at (i_k, j_{k-1}): If $\forall j : Q_{i_k,j} \neq 1^*$ all previous sequence elements are flipped, i.e. $1 \mapsto 1^*$, $1^* \mapsto 1$ and step 2 is restarted. Otherwise $((i_k, j_k), 1^*)$ with $Q_{i_k,j_k} = 1^*$ is recorded and case b takes effect.

b) A 1^* is found and recorded at (i_k, j_k): If $\forall i : Q_{i,j_k} \neq 1$, we record the row i_k as essential and remove the sequence elements $((i_k, j_k), 1^*)$ and $((i_k, j_{k-1}), 1)$, return to case b) and continue searching for a 1 in column j_{k-1} for rows $i_{k'} > i_k$. If a 1 is found at (i_{k+1}, j_k), check that $\forall k' < k + 1 : i_{k'} \neq i_{k+1}$. If true, record $((i_{k+1}, j_k), 1)$ and go to case a), else continue searching for a 1 in column j_k in rows $i_{k'} > i_{k+1}$.

For a flowchart representation of step 2, refer to section D in the Appendix.

3. This step needs a cover $\{u_i, v_j\}$ and a set of essential rows $\varepsilon_r \subset \{1, \cdots, n\}$ and columns $\varepsilon_c \subset \{1, \cdots, n\}$, where a column is considered essential if it contains a 1^* in an inessential row. Then compute

$$d = \min_{\substack{i \in \varepsilon_r^C \\ j \in \varepsilon_c^C}} u_i + v_j - R_{ij}$$

If there is no (i, j) such that $i \in \varepsilon_r^C, j \in \varepsilon_c^C$, then Q is a solution to the LAP. Otherwise, consider two cases:

a) $\forall i \in \varepsilon_r^C : u_i > 0$: With $m = \min_{i \in \varepsilon_r^C}(d, u_i)$ redefine:

$$\tilde{u}_i := \begin{cases} u_i & , i \in \varepsilon_r \\ u_i - m & , i \in \varepsilon_r^C \end{cases} \quad \tilde{v}_j := \begin{cases} v_j + m & , i \in \varepsilon_c \\ v_j & , i \in \varepsilon_c^C \end{cases}$$

b) $\exists i \in \varepsilon_r^C : u_i = 0$: With $m = \min_{j \in \varepsilon_c^C}(d, v_j)$ redefine:

$$\tilde{u}_i := \begin{cases} u_i + m & , i \in \varepsilon_r \\ u_i & , i \in \varepsilon_r^C \end{cases} \quad \tilde{v}_j := \begin{cases} v_j & , i \in \varepsilon_c \\ v_j - m & , i \in \varepsilon_c^C \end{cases}$$

Now, $\tilde{Q}$ can be computed for the cover $\{\tilde{u}_i, \tilde{v}_j\}_{i,j}$ and one continues in step 2.

This original implementation of the algorithm has a runtime complexity of $\mathcal{O}(n^4)$, but it can be modified to run in $\mathcal{O}(n^3)$ (Burkard et al., 2012), which we will use in practice.

Also note that in the original formulation of the problem, R is limited to be in $\mathbb{N}^{n \times n}$. The operations used in the algorithm are well defined on $\mathbb{R}_+$ as well, allowing

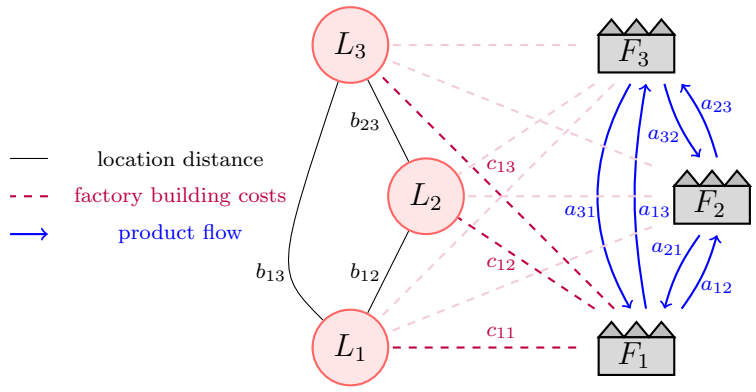

Figure 9: Visualization of the Quadratic Assignment Problem [GPT]

for $R \in \mathbb{R}_+^{n \times n}$. Lastly, any real-valued assignment problem can be transformed into an equivalent assignment problem with $\forall i, j : R_{ij} \geq 0$ (von Neumann, 1953), allowing us also to lift the requirement of non-negativity.

2.2.3. Quadratic Assignment Problem

Two years after *The Hungarian Algorithm* solving the LAP was presented, Koopmans and Beckmann (1957) introduced the Quadratic Assignment Problem (QAP). Instead of assigning jobs to workers, the setting is the construction of n facilities at n different locations. Hereby, it is assumed that the cost is proportional to the product of the flow from one facility to the next with the distance between them combined with the cost of building each facility at their respective locations. The QAP seeks to assign facilities to locations, minimizing the total cost. If this is encoded in matrices by letting a_{ik} be the flow from facility i to facility k, b_{jl} the distance between locations j and l and c_{ij} the cost of placing facility i at location j, we can define the QAP as follows:

Definition 5 (Quadratic Assignment Problem (Burkard et al., 2012))**.** *For three matrices $A, B, C \in \mathbb{R}^{n \times n}$ with $A = (a_{ik}), B = (b_{jl}), C = (c_{ij})$ and $\mathfrak{S}_n$ the symmetric group, the problem*

$$\arg\min_{\pi \in \mathfrak{S}_n} \left(\sum_{i=0}^{n-1} \sum_{k=0}^{n-1} a_{ik} b_{\pi(i)\pi(k)} + \sum_{i=0}^{n-1} c_{i\pi(i)} \right) \tag{31}$$

is called the Quadratic Assignment Problem (QAP) in Koopmans-Beckmann form, and equation 31 is called the QAP optimization function.

This problem has been used in a variety of scenarios. Past applications include the design of efficient typewriter layouts, campus planning, and turbine manufacturing (Burkard et al., 2012). Given this variety of perspectives, presenting the problem in another mathematically equivalent manner can be beneficial. By exploiting the following

properties:

$$\sum_{i=1}^{n}\sum_{k=1}^{n} a_{ik}b_{ik} = \operatorname{tr}(AB^\top)$$
$$(b_{\pi(i)\pi(k)}) = P_\pi B P_\pi^\top$$
$$\sum_{i=1}^{n} c_{i\pi(i)} = \operatorname{tr}(CP_\pi^\top)$$

where $P_\pi \in \Pi_n$ the permutation matrix corresponding to the permutation π, the proplem can be stated in its trace formulation (Burkard et al., 2012):

$$\arg\min_{P\in\Pi_n} \operatorname{tr}((APB^\top + C)P^\top) \tag{32}$$

Finding a global optimum for this problem is known to be NP-hard. A common technique is relaxing the constraints to the convex hull of Π_n, i.e.

$$\mathcal{D}_n = \bigcap_{\substack{\Pi_n \subset X \subset \mathbb{R}^{n\times n} \\ X \text{ convex}}} X$$

generally known as the Birkhoff Polytope or set of doubly-stochastic matrices, because they can also be defined to be $\mathcal{D}_n = \{D \in [0,1]^{n\times n}, \forall j \in \{1,\cdots,n\} : \sum_{i=1}^{n} D_{ij} = \sum_{i=1}^{n} D_{ji} = 1\}$. This modification allows us to use gradient descent to find a solution easily (Vogelstein et al., 2015) while still potentially providing (approximate) global solutions (Aflalo et al., 2015).

Definition 6 (Relaxed Quadratic Assignment Problem (Vogelstein et al., 2015))**.** *For three matrices $A, B, C \in \mathbb{R}^{n\times n}$ the problem associated with*

$$\min_{P\in\mathcal{D}_n} \operatorname{tr}((APB^T + C)P^T) \tag{33}$$

is called the Relaxed Quadratic Assignment Problem (rQAP), and equation 33 is referred to as the rQAP optimization function.

One can show that this problem is not necessarily convex, i.e., we can optimize towards local minima. To map a solution of rQAP to QAP, we project it to Π_n. An algorithm searching for a local minimum of rQAP in the special case of $C = 0$ is The Fast Approximate Quadratic Assignment Problem (FAQ) Algorithm (Vogelstein et al., 2015).

2.2.4. FAQ Algorithm

The FAQ algorithm (Vogelstein et al., 2015) is a gradient descent algorithm solving rQAP by making use of the Frank-Wolfe Algorithm (Frank and Wolfe, 1956) and the Hungarian Algorithm (Kuhn, 1955) and consists of three basic steps:

1. **Initialization:** The initial choice for $P^{(0)} \in \mathcal{D}_n$ can be arbitrary, but is generally chosen to be $P^{(0)} = \frac{1}{n}\mathbf{1}_n \cdot \mathbf{1}_n^\top$, the barycenter of the set.

2. **Solve:** We define the function $f(P)$ as the rQAP optimization function:

$$f(P) := -\text{tr}(APB^T P^T) \tag{34}$$

and apply the Frank-Wolfe algorithm (Frank and Wolfe, 1956) by iterating the following steps:

a) Compute the gradient of f at $P^{(i)}$: $\nabla f\left(P^{(i)}\right) = -AP^{(i)}B^\top - A^\top P^{(i)}B$

b) Compute the search direction $Q^{(i)}$ by using the first-order Taylor approximation of $f(P)$ around $P^{(i)}$

$$\tilde{f}^{(i)}(P) := f\left(P^{(i)}\right) + \nabla f\left(P^{(i)}\right)^\top (P - P^{(i)}) + \mathcal{O}(P^2) \tag{35}$$

and then, ignoring constant terms, solving:

$$Q^{(i)} := \underset{P \in \mathcal{D}_n}{\arg\min}\, \text{tr}\left(\nabla f\left(P^{(i)}\right)^\top P\right) \tag{36}$$

Recalling equation 25, this is a LAP and is solved exactly over a subset $\Pi_n \subset \mathcal{D}_n$ with the Hungarian Algorithm.

c) Compute the step size $\alpha^{(i)}$ by minimizing $f(P)$ along the line segment from $P^{(i)}$ to $Q^{(i)}$:

$$\alpha^{(i)} := \underset{\alpha \in [0,1]}{\arg\min}\, f(\alpha P^{(i)} + (1-\alpha)Q^{(i)}) \tag{37}$$

which can be solved exactly since it is a quadratic function in α.

d) $\alpha^{(i)}$ defines the doubly stochastic matrix for the next iteration:

$$P^{(i+1)} = P^{(i)} + \alpha^{(i)} Q^{(i)} \tag{38}$$

e) For a predefined stopping criterion ε, the steps a-d are repeated until the iteration limit is reached, $\left\|P^{(i)} - P^{(i-1)}\right\|_F < \varepsilon$ or $\left\|\nabla f(P^{(i)})\right\|_F < \varepsilon$, where $\|\cdot\|_F$ is the Frobenius Norm.

3. **Projection:** The final doubly stochastic matrix $P^{(n)}$ can be projected onto the Π_n using

$$\underset{P \in \Pi_n}{\arg\min} -\text{tr}(P^{(n)} P^T) \tag{39}$$

which is another instance of the LAP and can be solved with the Hungarian Alorithm in $\mathcal{O}(n^3)$.

Algorithm 1 Fast Approximate Quadratic Assignment Problem Algorithm (Vogelstein et al., 2015)

Require: $A, B \in \mathbb{R}^{n \times n}$

1: Initialize $P^{(0)} \leftarrow \frac{1}{n}\mathbf{1}_n \cdot \mathbf{1}_n^\top$
2: **for** $i = 1, 2, \cdots$ (while stopping criterion not met) **do**
3: Compute $\nabla f(P^{(i)}) = AP^{(i)}B^\top + A^\top P^{(i)}B$
4: $Q^{(i)} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \operatorname{tr}(Q^\top \nabla f(P^{(i)}))$ via Hungarian Algorithm
5: $\alpha^{(i)} \leftarrow \arg\max_{\alpha \in [0,1]} \alpha P^{(i)} + (1-\alpha)Q^{(i)}$
6: $P^{(i+1)} \leftarrow \alpha^{(i)} P^{(i)} + (1-\alpha^{(i)})Q^{(i)}$
7: **end for**
8: **return** $\hat{Q} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \operatorname{tr}(Q^\top P^{(\text{final})})$ via Hungarian Algorithm

Returning to the defining function of the GMP and using an alternative but equivalent definition of the Frobenius norm $\|U\|_F = \sqrt{\operatorname{tr}(U^\dagger U)}$ observe:

$$\begin{aligned}\left\|A_X - PA_ZP^T\right\|_F^2 &= \operatorname{tr}((A_X - PA_ZP^\top)^\dagger(A_X - PA_ZP^\top)) \\ &= \operatorname{tr}(A_X^\top A_X) + \operatorname{tr}(A_Z^\top A_Z) - 2 \cdot \operatorname{tr}(A_X P A_Z^\top P^\top)\end{aligned}$$

We find that equation 23 can be replaced with

$$\arg\min_{P \in \Pi} -\operatorname{tr}(A_X^\top P A_Z P^\top) = \arg\max_{P \in \Pi} \operatorname{tr}(A_X^\top P A_Z P^\top) \tag{40}$$

When the GMP is expressed using equation 40, the connection to the Quadratic Assignment Problem becomes apparent. Hence, the GMP can be approximately solved with the FAQ Algorithm.

2.2.5. Optimal Transport

Experiments show that FAQ performs very well on isomorphic graphs. In the more realistic case, at least in the context of BLI, two graphs are rarely isomorphic, and the performance decreases significantly. Additionally, due to the runtime complexity of $\mathcal{O}(n^3)$, the algorithm does not scale well for larger graphs (Saad-Eldin et al., 2021).

The main problem lies in step 2b, the approximation of the search direction, where the LAP is solved multiple times. Apart from the computational effort, the Hungarian Algorithm restricts the search space to Π_n. While this returns valid extrema, they need not be unique. For general LAP solvers, the tie-breaking process depends on the implementation and can be biased on the input, which is why it is shuffled in practice. For large inputs, this can again be computationally challenging. Furthermore, there is no objective reasoning in preferring one solution over another. It turns out that any convex combination of solutions to the LAP is also a solution.

Lemma 1 (Saad-Eldin et al. (2021)). *Let $M \in \mathbb{R}^{n\times n}$ be a matrix with n possible solutions to*

$$C = \max_{Q\in\Pi_n} \operatorname{tr}(Q^\top M)$$

where C is the optimal objective function value. Let $P_i, i \in \{0, 1, \cdots, n-1\}$ be those solutions and let

$$P = \sum_{i=0}^{n-1} \lambda_i P_i$$

a convex combination with $\lambda_i \in [0, 1]$ and $\sum_{i=0}^{n-1} \lambda_i$. Then P is a solution to

$$\arg\max_{Q\in\mathcal{D}_n} \operatorname{tr}(Q^\top M)$$

with $C = \operatorname{tr}(P^\top M)$

Proof. Since $\mathcal{D}_n$ is the convex hull of Π_n, any convex combination of permutation matrices is a doubly stochastic matrix, hence $P \in \mathcal{D}_n$. Furthermore,

$$\operatorname{tr}(P^\top M) = \operatorname{tr}\left(\sum_{i=0}^{n-1} \lambda_i P_i^\top M\right) = \sum_{i=0}^{n-1} \lambda_i \operatorname{tr}(P_i^\top M) = \sum_{i=0}^{n-1} \lambda_i C = C$$

□

Example 1 (Marchisio et al. (2022)). *Take the matrix M*

$$M = \begin{pmatrix} 0 & 3 & 0 \\ 2 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix}$$

Computing $\arg\max_{P\in\Pi_n} \operatorname{tr}(P^\top M)$ returns two possible solutions:

$$P_1 = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} \qquad P_2 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with $\operatorname{tr}(P_1^\top M) = \operatorname{tr}(P_2^\top M) = 5$. Instead of having to choose P_1 or P_2, it is preferable, intuitively, to choose the "direct path":

$$P_{\text{avg}} = \begin{pmatrix} 0 & 1 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} \end{pmatrix}$$

As proven in Lemma 1, the convex combination preserves the value: $\operatorname{tr}(P_{\text{avg}}^\top M)$.

To attempt step 2b deterministically while balancing the possible permutation matrix solutions, Saad-Eldin et al. (2021) proposed utilizing the optimal transport problem (OT).

Definition 7 (Optimal Transport Problem (Saad-Eldin et al., 2021))**.** *For the transportation polytope*

$$U(r,c) := \left\{P \in \mathbb{R}_+^{n\times n}, P\mathbf{1}_n = r, P^\top \mathbf{1}_n = c\right\} \tag{41}$$

with $\mathbf{1}_n = [1]^n$ *and* $r, c \in \mathbb{R}^n$ *and a cost matrix* $M \in \mathbb{R}^{n\times n}$*, the Optimal Transport Problem is defined as:*

$$\underset{P\in U(r,c)}{\arg\min}\, \mathrm{tr}(P^\top M) \tag{42}$$

or equivalently

$$\underset{P\in U(r,c)}{\arg\min} \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} M_{ij}P_{ij} \tag{43}$$

Once again, this class of problems originates in an economic setting and can be intuitively understood as a logistics problem (Solomon, 2017). Let there be m factories R_i producing an amount $r_i \geq 0, i \in \{0, \cdots, m-1\}$ of a good and n consumers C_j requesting an amount $c_j \geq 0, j \in \{0, \cdots, n-1\}$. Further consider the cost matrix M, where M_{ij} quantifies the cost of transport for one unit of the good from producer R_i to consumer C_j. The entries of the planning matrix (P_{ij}) represent the amount of the goods transported from facility R_i to consumer C_j. The cost for this transaction can be calculated as $M_{ij}P_{ij}$. Then, it is clear the equation 43 represents the total transport cost for all transactions. There are still some boundary conditions necessary:

- A facility ships as much as it produces:

$$\forall i : \sum_{j=0}^{n-1} P_{ij} = r_i \tag{44}$$

- A consumer only receives the amount he requests:

$$\forall j : \sum_{i=0}^{n-1} P_{ij} = c_j \tag{45}$$

- Items are shipped from facilities to consumers

$$\forall i, j : P_{ij} \geq 0 \tag{46}$$

Nevertheless, in this formulation, the OT is just a generalization of the LAP (choose

$r = c = 1$), which can still be computationally prohibitive to solve for larger supports. To combat this, Cuturi (2013) suggested adding an entropic regularization term to smooth the conventional OT.

Definition 8 (Optimal Transport Problem (Sinkhorn Distance Formulation) (Cuturi, 2013))**.** *Let the entropy $h : \mathbb{R}_+^{n\times n} \to \mathbb{R}$ be defined as*

$$h(P) := -\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} P_{ij}\log(P_{ij}) \tag{47}$$

For $\lambda > 0$ and a cost matrix $M \in \mathbb{R}^{n\times n}$ the optimal transport problem in the sinkhorn distance formulation is defined as

$$P^\lambda = \underset{P\in U(r,c)}{\arg\min}\,\mathrm{tr}(P^\top M) - \frac{1}{\lambda}h(P) \tag{48}$$

Note, for $\lambda \to \infty$, the sinkhorn distance coincides with the solution of the OT. Cuturi (2013) further showed the following lemma:

Lemma 2 (Cuturi (2013))**.** *For $\lambda > 0$, the solution P^λ is unique and has the form $P^\lambda = \mathrm{diag}(u)K\mathrm{diag}(v)$ where $u, v \in \mathbb{R}_+^n$ are unique up to scaling and $K := \exp(-\lambda M)$ with $\exp(\cdot)$ the element-wise exponential.*

As a consequence, P^λ can be computed using the Sinkhorn-Knopp fix-point iteration algorithm by alternately normalizing the rows and columns of K, which guarantees convergence if K has total support (Knopp and Sinkhorn, 1967).

Definition 9 (Total Support (Knopp and Sinkhorn, 1967))**.** *For a square matrix $A \in \mathbb{R}^{n\times n}, A \neq 0$ and a permutation $\pi \in \mathfrak{S}_n$ the elements $A_{1\pi(1)}, \cdots, A_{n\pi(n)}$ are called the diagonal of A corresponding to π. If $\forall i : A_{i\pi(i)} > 0$, this diagonal is called positive. A matrix is called to have total support if every positive element of A lies on a positive diagonal.*

Using this definition and since $\exp(-\lambda M)$ only has positive entries (i.e., has total support) for any M, the algorithm converges for any input $K = \exp(-\lambda M)$. For $r = c = \mathbf{1}_n$, we recover the Birkhoff Polytope and refer to this special case as the *Doubly Stochastic Optimal Transport Problem.*

Definition 10 (Doubly Stochastic Optimal Transport Problem (Saad-Eldin et al., 2021))**.** *For $r = c = \mathbf{1}_n$, the OT is called the Doubly Stochastic Optimal Transport Problem and solves for*

$$\underset{P\in\mathcal{D}_n}{\arg\min}\,\mathrm{tr}P^T M \tag{49}$$

which corresponds to a relaxed version of equation 25, the Linear Assignment Problem. Hence, the Sinkhorn-Knopp algorithm can be used to solve it.

Algorithm 2 Sinkhorn-Knopp Algorithm (Knopp and Sinkhorn, 1967)

Require: Matrix $K \in \mathbb{R}^{n \times n}$ with total support, constraints $r, c \in \mathbb{R}^n$

Initialize $u \leftarrow \mathbf{1}_n$ (vector of ones of dimension n)

repeat

  $v \leftarrow \frac{c}{K^\top u}$ ▷ Ensures column constraints

  $u \leftarrow \frac{r}{Kv}$ ▷ Ensures row constraints

until Convergence or Maximum Iterations

Compute $P \leftarrow \text{diag}(u) \cdot K \cdot \text{diag}(v)$

return P

Algorithm 3 Lightspeed Optimal Transport Algorithm (Saad-Eldin et al., 2021)

Require: Cost Matrix $M \in \mathbb{R}^{n \times n}$, $\lambda \in \mathbb{R}$

Compute $K \leftarrow \exp(-\lambda M)$

Compute $P \leftarrow \text{Sinkhorn} - \text{Knopp}(K)$

return P

In practice it suffices to use $\lambda \geq 100$. Cuturi (2013) has shown that the Sinkhorn-Knopp Algorithm has an empirical runtime complexity of $\mathcal{O}(n^2)$ - the potential to speed up the FAQ-Algorithm.

2.2.6. Graph Matching via Optimal Transport

Replacing the Hungarian Algorithm with the Lightspeed Optimal Transport Algorithm (LOT) in the FAQ Algorithm results in the Graph Matching via Optimal Transport Algorithm (GOAT). As with the FAQ Algorithm, any valid doubly stochastic matrix

Algorithm 4 Graph Matching via Optimal Transport (Saad-Eldin et al., 2021)

Require: $A, B \in \mathbb{R}^{n \times n}$

Initialize $P^{(0)} \leftarrow \frac{1}{n}\mathbf{1}_n \cdot \mathbf{1}_n^\top$

for $i = 1, 2, \cdots$ (while stopping criterion not met) **do**

  Compute $\nabla f(P^{(i)}) = AP^{(i)}B^\top + A^\top P^{(i)}B$

  $Q^{(i)} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \text{tr}(Q^\top \nabla f(P^{(i)}))$ via LOT

  $\alpha^{(i)} \leftarrow \arg\max_{\alpha \in [0,1]} \alpha P^{(i)} + (1-\alpha)Q^{(i)}$

  $P^{(i+1)} \leftarrow \alpha^{(i)} P^{(i)} + (1-\alpha^{(i)})Q^{(i)}$

end for

return $\hat{Q} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \text{tr}(Q^\top P^{(\text{final})})$ via Hungarian Algorithm

can be chosen as initialization. In a direct comparison of FAQ and GOAT, one finds that GOAT outperforms FAQ in terms of speed and performance in graph matching on non-isomorphic, large ($n > 1500$) graphs (Saad-Eldin et al., 2021). This can be intuitively understood by the following visualization:

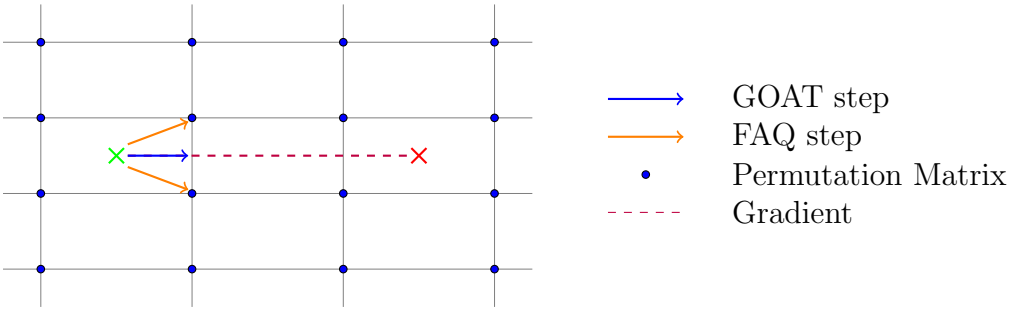

Figure 10: Visualization of optimization step in GOAT vs. FAQ. Adapted from Marchisio et al. (2022) [GPT]

2.2.7. Seeded Graph Matching

Up to now, we have considered problems for which no additional information is available. In reality, the setting is usually a bit better. For example, in the context of BLI, a native speaker might be available to provide a couple of translations, which can be leveraged as seeds. The naturally resulting question is whether this information can improve our efforts in graph matching. This variation is known as the Seeded Graph Matching Problem.

Definition 11 (Seeded Graph Matching (Fishkind et al., 2019))**.** *Let G_X, G_Z be two graphs with their respective vertex sets V_X, V_Z, such that $|V_X| = |V_Z|$, subsets $W_X \subset V_X, W_Z \subset V_Z$, such that $|W_X| = |W_Z|$ and a fixed bijection $\varphi : W_X \to W_Z$. The seeded graph matching problem (SGM) is defined to minimize the edge disagreements induced by ϕ over all bijections $\phi : V_X \to V_Z$, with $\phi|_{W_X} = \varphi$. The elements of W_X, W_Z are called seeds and φ a seeding.*

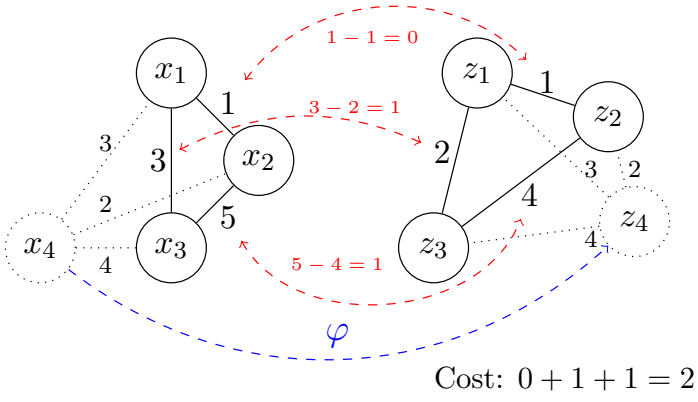

Figure 11: Visualization of Seeded Graph Matching. Adapted from Marchisio et al. (2021) [GPT]

Equivalently, let $V_X = V_Z = \{1, 2, \cdots, l\}$ with $l = m + n$ and seeds $W_X = W_Z = \{1, 2, \cdots, m\}$. Without loss of generality, let the seeding $\varphi = \mathrm{Id}_m$ be the identity. Also,

assume a partitioning of the adjacency matrices $A, B \in \mathbb{R}^{l \times l}$:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \qquad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}$$

with $A_{11}, B_{11} \in \mathbb{R}^{m \times m}$, $A_{12}, B_{12} \in \mathbb{R}^{m \times n}$, $A_{21}, B_{21} \in \mathbb{R}^{n \times m}$ and $A_{22}, B_{22} \in \mathbb{R}^{n \times n}$. The SGM can then also be expressed using

$$\arg\min_{P \in \Pi_n} \left\| A - (\mathbb{1}_m \oplus P) B (\mathbb{1}_m \oplus P)^T \right\|_F^2 \tag{50}$$

Analogously to the derivation of equation 40 we get the equivalent formulation:

$$\arg\max_{P \in \Pi_n} \operatorname{tr}\left(A^\top (\mathbb{1}_m \oplus P) B (\mathbb{1}_m \oplus P^\top) \right) \tag{51}$$

We can now relax the search space again and define the *Relaxed Seeded Graph Matching Problem* (rSGM) (Fishkind et al., 2019) via

$$\arg\max_{P \in \mathcal{D}_n} \operatorname{tr}\left(A^\top (\mathbb{1}_m \oplus P) B (\mathbb{1}_m \oplus P^\top) \right) \tag{52}$$

This special case of the (relaxed) GMP allows us to apply FAQ (GOAT) when solving it. The objective function, in this case, is defined as

$$\begin{aligned} f(P) &= \operatorname{tr}\left(\begin{pmatrix} A_{11}^\top & A_{12}^\top \\ A_{21}^\top & A_{22}^\top \end{pmatrix} \begin{pmatrix} \mathbb{1}_m & 0_{m \times n} \\ 0_{n \times m} & P \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} \begin{pmatrix} \mathbb{1}_m & 0_{n \times m} \\ 0_{m \times n} & P^\top \end{pmatrix} \right) \\ &= \operatorname{tr}\left(\begin{pmatrix} A_{11}^\top & A_{12}^\top \\ A_{21}^\top & A_{22}^\top \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} P^\top \\ P B_{21} & P B_{22} P^\top \end{pmatrix} \right) \\ &= \operatorname{tr}\left(A_{11}^\top B_{11} \right) + \operatorname{tr}\left(P^\top A_{21} B_{21}^\top \right) + \operatorname{tr}\left(P^\top A_{12}^\top B_{12} \right) + \operatorname{tr}\left(A_{22} P B_{22} P^\top \right) \end{aligned} \tag{53}$$

The gradient is then computed as

$$\nabla f(P) = A_{21} B_{21}^\top + A_{12}^\top B_{12} + A_{22} P B_{22}^\top + A_{22}^\top P B_{22} \tag{54}$$

The Seeded Graph Matching Algorithm (Fishkind et al., 2019) corresponds to running the FAQ algorithm with equation 53 as objective.

A concluding note on the space and time complexity - in both GOAT and FAQ, the gradient matrix that is computed will be dense, resulting in a space complexity of $\mathcal{O}(n^2)$ (Saad-Eldin et al., 2021). FAQ and GOAT also share the time complexity class of $\mathcal{O}(n^3)$, even though GOAT has an order of magnitude minor polynomial coefficient terms. This effect is only noticeable for larger graphs (Saad-Eldin et al., 2021). Subsequently, the same space complexity is also held by SGM, where n is the number of non-seed vertices. Furthermore, unless the number of seeds is excessive, they barely influence the runtime in practice, leaving it also with a runtime complexity of $\mathcal{O}(n^3)$ (Fishkind et al., 2019).

Algorithm 5 Seeded Graph Matching Algorithm (Fishkind et al., 2019)

Require: Graphs G_X, G_Z with vertex sets $\{1, 2, \cdots, m+n\}$, with respective adjacency matrices $A, B \in \mathbb{R}^{m+n \times m+n}$, assuming vertices $\{1, \cdots, m\}$ are seeds

Initialize $P^{(0)} \leftarrow \frac{1}{n}\mathbf{1}_n \cdot \mathbf{1}_n^\top$
for $i = 1, 2, \cdots$ (while stopping criterion not met) **do**
    Compute $\nabla f(P^{(i)}) = A_{21}B_{21}^\top + A_{12}^\top B_{12} + A_{22}P^{(i)}B_{22}^\top + A_{22}^\top P^{(i)}B_{22}$
    $Q^{(i)} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \operatorname{tr}(Q^\top \nabla f(P^{(i)}))$ via Hungarian Algorithm (or LOT)
    $\alpha^{(i)} \leftarrow \arg\max_{\alpha \in [0,1]} \alpha P^{(i)} + (1-\alpha)Q^{(i)}$
    $P^{(i+1)} \leftarrow \alpha^{(i)}P^{(i)} + (1-\alpha^{(i)})Q^{(i)}$
end for
return $\hat{Q} \leftarrow \arg\max_{Q \in \mathcal{D}_n} \operatorname{tr}(Q^T \nabla f(P^{(\text{final})})$ via Hungarian Algorithm

2.3. Isomorphism Hypothesis

A subtle but essential assumption that was made in the formulation of equation 12 can cause issues for distant languages:

$$\arg\min_{W \in \mathbb{R}^{d \times d}} \sum_{i=0}^{n-1} \|Wx_i - z_i\|_2^2$$

This assumption is called the *Isomorphism Hypothesis* (sometimes *Isometry Hypothesis*) and assumes that the embedding spaces of the source and target language are roughly isomorphic. It has been shown that this assumption weakens with increasing etymological distance between the source and target languages (Søgaard et al., 2018; Patra et al., 2019). First, some mathematical background:

Definition 12 (Isomorphism (Vector Spaces) (Bosch, 2014))**.** *For a field $\mathbb{K}$, a $\mathbb{K}$-linear mapping $f : V \to W$ between two $\mathbb{K}$ vector spaces V, W is called an isomorphism if the mapping is injective and surjective, i.e., it holds:*

$$\forall x, y \in V : f(x) = f(y) \implies x = y \qquad \forall y \in W \exists x \in V : f(x) = y \tag{55}$$

The two vector spaces are called isomorphic if such an isomorphism exists.

If we now recall the exact formulation of the Procrustes Problem, we are not dealing with general linear mappings W but restrict the search space to the space of orthogonal matrices. In this case, we can call the mapping an *isometry*.

Definition 13 (Isometry (Bosch, 2014))**.** *For a field $\mathbb{K}$, a $\mathbb{K}$-linear mapping $f : V \to V$, where V is a $\mathbb{K}$ vector space with a scalar product $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{K}$, is called in isometry if it holds that:*

$$\forall x \in V : |f(x)| = |x| \tag{56}$$

for the metric $|x| := \sqrt{\langle x, x \rangle}$. If $\mathbb{K} = \mathbb{R}$, f is called an orthogonal mapping.

Returning to the BLI use case, it would be helpful to quantify the degree of isometry between two vector spaces. This can indicate how well the spaces can be aligned and how well the learned mapping can perform. A possible metric is the Gromov-Hausdorff distance (GH). It is based on the Hausdorff distance, which intuitively measures the distance between the most distant nearest neighbors and is given by (Patra et al., 2019):

$$H(V,W) = \max\{\sup_{v\in V}\inf_{w\in W} d(v,w), \sup_{w\in W}\inf_{v\in V} d(v,w)\} \tag{57}$$

where V, W are metric spaces, which holds for $V = \mathbb{R}^n$, and $d(\cdot,\cdot)$ the corresponding distance function. The Gromov-Hausdorff distance computes the minimal Hausdorff distance taken over all isometric transforms $f : V \to Z,\ g : W \to Z$ into a shared compact metric space (Z, d), where $V, W \subset Z$ are closed subsets of Z (Patra et al., 2019; Mémoli, 2008):

$$\mathcal{H}(V,W) = \inf_{f,g} H(f(V), g(W)) \tag{58}$$

A similar problem can arise when changing to a graph-based perspective. The definition of isomorphy in graphs is slightly different. Let us again start with a basic introduction to Graph Theory:

Definition 14 (Graph (Balakrishnan and Sridharan, 2018; Mathew et al., 2023))**.** *An undirected graph G is a tuple (V, E), where V is a finite, non-empty set of vertices and $E \subset V \times V$ is a set of unordered pairs (i, j) of vertices $i, j \in V$. The elements of E are called edges. The graph is called weighted if each edge $e \in E$ is assigned a weight $w(e)$. We will denote weighted graphs as (V, E, w). The degree of a vertex $d(i)$ corresponds to the number of edges containing i, i.e. $d(i) = |\{(k, l) \in E | k = i \vee l = i\}|$.*

If the elements of E are ordered pairs (i, j) of vertices $i, j \in V$, the elements are called arrows, and G is called directed.

The corresponding adjacency matrix for unweighted graphs is defined as $A \in \{0, 1\}^{n\times n}$, with $(A)_{ij} = 1 \iff \{i, j\} \in E$.

Given these definitions, we can now introduce the notion of isomorphy for graphs:

Definition 15 (Isomorphism (Graphs) (Balakrishnan and Sridharan, 2018))**.** *Let $G_X = (V_X, E_X), G_Z = (V_Z, E_Z)$ be two graphs, G_X is called isomorphic to G_Z, if there exists a bijective mapping $\iota : V_X \to V_Z$, such that it holds:*

$$\forall i, j \in V_X : (i, j) \in E_X \iff (\iota(i), \iota(j)) \in E_Z \tag{59}$$

If such a mapping exists, G_X and G_Z are called isomorphic.

In the context of BLI, we usually deal with weighted graphs, which capture the similarity of connected nodes. Our definition of graph isomorphism does not consider weights, so

we can weaken the requirement by checking whether the underlying 1-nearest neighbor subgraphs for the l most frequent words are isomorphic (Søgaard et al., 2018).

Definition 16 (k-Nearest Neighbor Graph (Dong et al., 2011))**.** *For a set of objects V, the k-nearest neighbor graph is a directed graph with vertex set V and an edge, from each $v \in V$ to its k most similar objects in V, for a given similarity measure, i.e.*

$$E = \{(i, j) | i, j \in V : j \in \mathcal{N}_k(i)\} \tag{60}$$

where $\mathcal{N}_k(i)$ is the set of k nearest neighbors of i for a given metric.

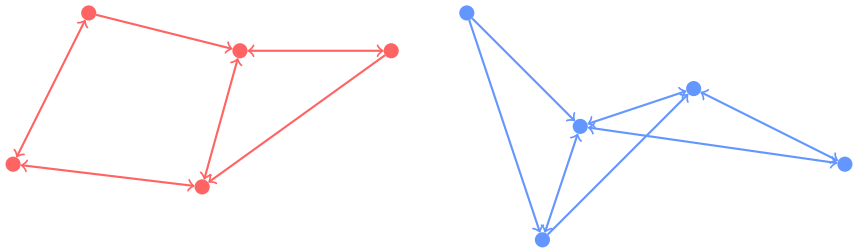

Figure 12: Visualization of two non-isomorphic k-Nearest Neighbor Graphs for $k = 2$ with the Euclidean metric. [GPT]

Similar to the vector space setting, we want to find another metric that can reflect the degree of isomorphism for graphs. Søgaard et al. (2018) present a metric based on the similarity of Laplacian eigenvectors. For a graph G, they are computed as $L = D - A$, where D is a diagonal matrix containing the degrees of the vertices and A is the corresponding adjacency matrix. We can now define the eigensimilarity Δ of the Laplacians L_X, L_Z of the 1-nearest neighbor graphs as (Søgaard et al., 2018):

$$\Delta = \sum_{i=0}^{l-1} (\lambda_{X_i} - \lambda_{Z_i})^2 \tag{61}$$

where l is chosen as the smallest l such that the sum of the l largest eigenvalues of the Laplacians is $< 90\%$ of the total sum, i.e.:

$$l = \min_{j \in \{X,Z\}} \arg\min_{k \in \mathbb{N}} \frac{\sum_{i=0}^{k-1} \lambda_{j_i}}{\sum_{i=0}^{n-1} \lambda_{j_i}} > 0.9 \tag{62}$$

Both of the presented metrics GH and Δ decrease with increasing graph similarity and show correlations with the performance achieved in BLI (Søgaard et al., 2018; Patra et al., 2019).

2.4. Data Sources

Generally, there aren't many dictionaries available for the Oshiwambo dialects. But there has been a recent development for Oshikwanyama, where the Oshikwanyama Curriculum

Committee (2019) compiled a dictionary based on two early dictionaries from 1954 (Tobias, 1954) and 1977 (Turvey et al., 1977). The 1954 version already illustrates the problems that can arise when working with Oshikwanyama. Since the first official Oshikwanyama orthography was only published in 1968 (Upgrading African Languages Project Namibia, 2004), the earlier dictionary cannot comply with the orthographic rules of modern Oshikwanyama. Furthermore, according to a master's thesis on the subject, as one might expect, much of the dictionary's content is outdated (Shikesho, 2019). It is also worth mentioning the early work of Hermann Tönjes, who created a German-Oshikwanyama dictionary (Tönjes, 1910b) as well as a textbook (Tönjes, 1910a) in 1910, although neither is helpful for reasons mentioned above.

The situation is a bit better for dictionaries covering Oshindonga since there exist younger books, from 1984 (Viljoen et al., 1984), 1986 (Tirronen, 1986) and 1996 (ELCIN Church Council Special Committees, 1996).

Due to Namibia's history, the bible has been translated into Oshikwanyama and Oshindonga. It is digitally provided by YouVersion[3]. We decided against using this as the basis for the thesis for two reasons. When previously working with the Oshikwanyama translation of the bible, it was difficult to accurately determine which bible version was used. Finding the corresponding English version is impossible without this information, leading to problems. The other major drawback in using the bible is that the generated dictionary would not contain any contextual modern words, one of the goals this thesis is trying to achieve.

Another option could be news articles. Namibia has one major Newspaper called The Namibian[4], which publishes articles in English as well as Oshiwambo. The main advantage is that it provides up-to-date vocabulary on various topics. Unfortunately, The Namibian does not discriminate between Oshikwanyama and Oshindonga articles, which would result in the Oshiwambo corpus containing two dialects.

Lastly, the group *Masakhane NLP* created the Writing Our Narratives (WON) data set in the *AfricaNLP* Workshop at *ICLR 2022* (Nekoto et al., 2022). For this, they invited 11 highly proficient participants in English and Oshindonga and asked them to develop and translate sentences addressing various topics. This resulted in a collection containing 5419 relatively short sentences. In their machine translation project, they additionally created and used a parallel corpus of the Oshikwanyama translation of the Namibian constitution and were granted access to a third data source consisting of 1000 questions and answers around touristic bookings which had also been translated to Oshindonga.

[3]https://www.bible.com
[4]https://namibian.com.na

2.5. Prior Work on the Oshiwambo Language

To my knowledge, only one other group has attempted BLI for Oshiwambo (Nakashole, 2019). Their approach is based on finding a projection matrix that maps the source language's word embeddings to the target language's corresponding word embeddings as introduced by Mikolov et al. (2013a). This approach assumes the two embedding vector spaces to be isomorphic and scores $0.30, 0.56$ and 0.58 in precision at top-k ($P@k$) for $k \in \{1, 5, 10\}$ respectively for the generated English-Oshikwanyama language pair. Interestingly, their results for English-Italian are similar with values of $0.34, 0.48$ and 0.54 in $P@k$ for $k \in \{1, 5, 10\}$ respectively. However, experiments have shown that the isomorphism assumption is less sensible in cases where the language similarity decreases or in cases of varying domains of the monolingual training data sets (Søgaard et al., 2018; Patra et al., 2019). In similar languages like English and Spanish, it has been established that a bigger training dataset results in a better performance (Mikolov et al., 2013a). This raises the question of whether the same can be expected in our case of Oshiwambo and if the results are scalable to larger generated dictionary sizes.

There has also been an attempt at Neural Machine Translation by training new and fine-tuning existing translation models (Nekoto et al., 2022). For this process, a new dataset containing 5000 sentences has been established based on the Namibian constitution and data created by bilingual speakers as part of a workshop. The group achieved a BLEU score of 8.9 when training a model from scratch and a 24.2 BLEU score as the overall best result by fine-tuning the multilingual M2M-100 model.

3. Methodology

The following section is divided into three subsections, chronologically introducing three of the four necessary steps in creating the Oshiwambo-English dictionary. We start by justifying our choice of data and introducing the essential preprocessing steps. Next, we will present the BytePair Embeddings in more detail before we finish the chapter with an overview of the algorithms used to solve the Procrustes and Graph-Matching Problems. The final step, the evaluation, will be explained in a separate chapter.

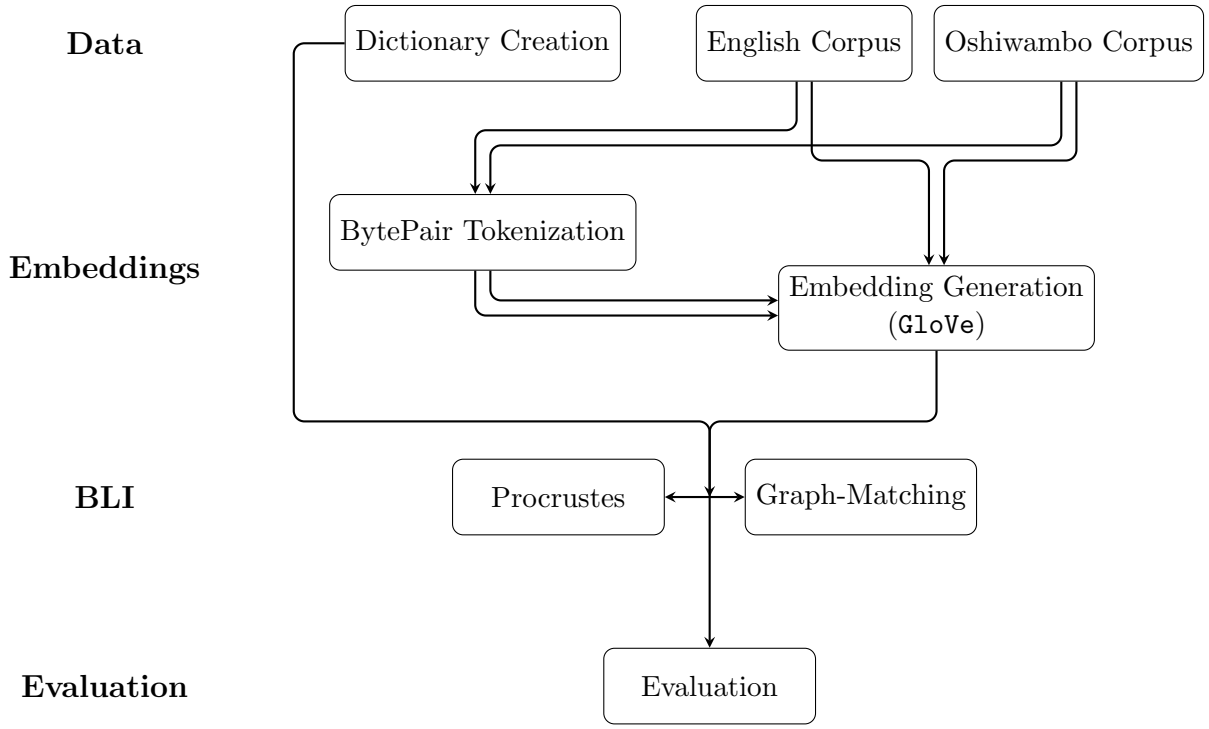

Figure 13: Overview of the four elemental steps in creating an Oshiwambo-English dictionary using BLI. [GPT]

Inspired by the idea of using a related language with more available resources as an intermediary (Sannigrahi and Read, 2022), I set up a similar experiment for Swahili, another representative of the Bantu languages as the target language. Since English and Swahili are official languages in Tanzania (Mascarenhas, 2025), substantially more resources are available. Additionally, Swahili is supported by `Google Translate`, facilitating the evaluation. I hope to gain insights into the corpus size dependency with this second experiment. Additionally, I want to investigate the possibility of merging the Oshiwambo and Swahili corpora to generate more robust embeddings, potentially improving the performance of Oshiwambo-English BLI.

3.1. Data Acquisition

One of the main criteria for a possible data source in the selection process was its bilingual availability since using two corpora from varying domains could drastically reduce the performance (Søgaard et al., 2018). In the case of Oshiwambo, newspapers proved to be the most valuable asset. Given a bilingual publisher, you do not have the problem of dissimilar domains, as creating a parallel corpus from the corresponding articles is possible. Using the `Fundus` library (Dallabetta et al., 2024) I crawled The Namibian[5] for Oshiwambo-English and Daily News[6] and Habari Leo[7] for Swahili - English. An unavoidable drawback to using The Namibian as a data source is the imbalance between the resulting English and Oshiwambo corpora sizes. This asymmetry is accepted since more articles are published in English, and the Oshiwambo articles are not linked to their English correspondences. Not arbitrarily removing English articles guarantees the existence of a translation for each Oshiwambo article.

Additionally, the Namibian constitution is also available in English (von Wietersheim, 2018) and Oshindonga (von Wietersheim and Ndinoshiho, 2017). The text from the constitution and the plaintexts of the articles are combined into one file, lower-cased, stripped of non-letter characters, and all numbers replaced by zeroes.

This process generates an Oshiwambo corpus containing 45,038 sentences and a Swahili corpus with 130,053 sentences, which is short compared to the corpora used by Marchisio et al. (2022). We will refer to this corpus as the small corpus. This is why a second corpus is introduced, merging the Oshiwambo with the Swahili corpus and combining the English corpora generated from Namibian and Tanzanian sources. This corpus is used in the experiments to determine whether extending the corpora of low-resource languages with higher-resource, related languages can be beneficial and is denoted as the extended corpus.

Both the Procrustes and the graph-matching algorithm require seed translations as input. These are retrieved from existing dictionaries (Tirronen, 1986; ELCIN Church Council Special Committees, 1996). Due to their age, the dictionaries are only available as hard copies and scanned with optical-character-recognition-capable scanners. The text can then be extracted computationally, from which, in turn, the translations are retrieved using regular expressions. This technique is error-prone but is still feasible since we can limit seeds to words occurring in the source and target corpora, most likely discarding any parsing errors. Since only the extracted translations from the 1989 dictionary are of adequate quality, the 1998 dictionary is abandoned.

The seeds are generated by saving the n most frequent words of the Oshiwambo and English corpora in frequency-ordered lists L_O, L_E. Each word $w^X \in L_O$ in the Oshiwambo list has an entry in the dictionary; each possible translation w^Z is tested to be in the

[5]https://www.namibian.com.na/
[6]https://dailynews.co.tz/
[7]https://habarileo.co.tz/

English list. For each w^X with a translation $w^Z \in L_E$, a tuple (i_O, i_E), with i_O, i_E such that $L_O(i_O) = w^X$ and $L_E(i_E) = w^Z$ is returned as a seed. With the given corpus and $n = 3000$, we can generate such 214 seeds from scans of the 1989 dictionary. Additionally, due to the dissimilarity of English and Oshiwambo, all words $x \in L_O \cap L_E$ with $|w^X| \geq 4$ can be assumed to be names and are used as further input seeds. The full list of names can be found in section B.

Acquiring the seeds in Swahili is more straightforward since the language is supported by `Google Translate`[8]. Similarly, as before, the $n = 3000$ most common words in the Swahili and English corpora are compiled as a list and then translated using `Google Translate`. This dictionary can be utilized as input seeds and for automatic evaluation.

3.2. Embeddings

In this thesis, some concepts introduced in section 2 are combined to find a more performant approach to BLI for Oshiwambo. The first choice to be made is the type of embeddings used. It is natural to choose a kind of subword embedding, for it has the potential to better reflect relationships between words like *Namibia* and *moNamibia* (in Namibia). Due to the unavailability of a comparable resource like the Swahili Syllabic Alphabet (Shikali et al., 2019), we chose to use Byte Pair embeddings (Sennrich et al., 2016).

The Byte-Pair embeddings were trained using the training scripts provided in the GitHub repository[9] corresponding to the publication (Heinzerling and Strube, 2018). In the first step, the corpus is tokenized and encoded using `SentencePiece` (Kudo and Richardson, 2018). The embeddings are then trained on the encoded corpora using `GloVe` (Pennington et al., 2014). In the final step, the embeddings must be re-arranged in the same order as in the BPE vocabulary file generated in the tokenization phase.

Both of the approaches require word embeddings. A given word is encoded using the learned BPE model. The corresponding embedding is computed by calculating the (weighted) arithmetic mean of the subword embeddings. Given a word w^X with a tokenization $(t_1, \cdots, t_n)$. Let τ_i be the embedding corresponding to the token t_i, then the mean word embeddings are computed by:

$$x = \frac{1}{n} \sum_{i=1}^{n} \tau_i \tag{63}$$

The weighted mean embeddings are calculated using the token length $|t_i|$ as weights.

$$x = \sum_{i=1}^{n} \frac{1}{|t_i|} \tau_i \tag{64}$$

[8] https://translate.google.com/
[9] https://github.com/bheinzerling/bpemb

The previously covered embeddings were embeddings associated with a fixed word. When attempting translation, it is not guaranteed that all encountered words are present in the training data. Sennrich et al. (2016) presented the Byte Pair Embedding approach to investigate the benefits of sub-word embeddings. If successful, the benefits for machine translation are obvious: composed out-of-vocabulary words could be translated based on the translations of the sub-words.

As the name indicates, these embeddings are based on the Byte Pair encoding (Gage, 1994). A new, unused byte iteratively replaces the most frequent pair of bytes for a given string of bytes. The algorithm works as follows. First, the symbol vocabulary is initialized with the characters present in the corpus and an end-of-word symbol '.'. Next, iteratively, all symbol pairs are counted, and the most frequent ones are replaced: ('A', 'B') $\rightarrow$ 'AB'. This is called a *merge operation*, and the total amount o is the only hyperparameter of the algorithm. Each merge operation produces a character n-gram, which is added to the symbol vocabulary (Sennrich et al., 2016).

Algorithm 6 Byte Pair Tokenization (Sennrich et al., 2016)

Require: corpus, o
vocab $\leftarrow$ get_all_characters(corpus)
for $i = 1, 2, \cdots, o$ (for o merge operations) **do**
    pair $\leftarrow$ get_most_frequent_symbol_pair(corpus)
    new_symbol $\leftarrow$ combine(pair)
    vocab $\leftarrow$ vocab.add(new_symbol)
    corpus $\leftarrow$ corpus.replace(pair, new_symbol)
end for
return vocab, corpus

Once the corpus is encoded, the embeddings can be trained using, e.g., `GloVe`.

3.3. Bilingual Lexicon Induction

Generally, Marchisio et al. (2022) have shown that the best results are achieved using a combination of Procrustes and graph matching. Choosing the correct algorithm to start with (heavily) depends on the number of seeds used. Given the moderate amount of seeds available, starting with graph matching has proven to be the better choice, which is why the following introduction will follow this order.

First, graph matching is run in the forward and reverse directions. For each direction a hypothesized mapping $h_i : \{1, n\} \rightarrow \{1, n\}, i \in \{f, b\}$ is returned. The indices f, b represent forward and backward, respectively. The hypothesis sets for each direction can then be defined as $H_f = \{(a, b)|h_f(a) = b\}$ and $H_b = \{(a, b)|h_b(b) = a\}$. The intersection $H = H_f \cap H_b$ is used as an input for Procrustes.

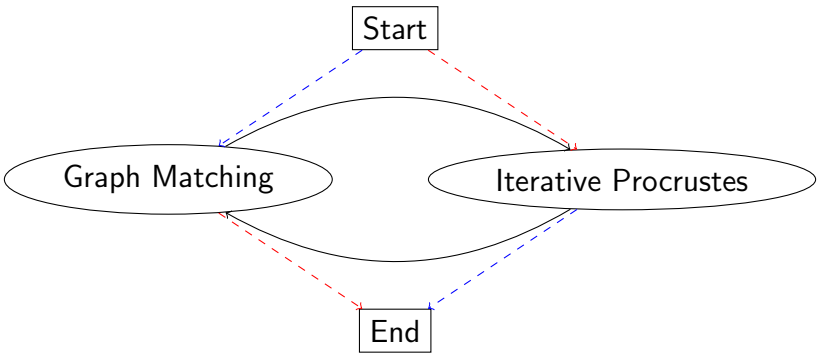

Figure 14: Overview of the system combination experimental setup. Adapted from Marchisio et al. (2022) [GPT]

Similarly, the Procrustes problem is solved in both directions using the gold seeds and the hypotheses from the graph-matching approach. The hypotheses are extracted using 1-nearest neighbors. Instead of the commonly used cosine metric, we will use Cross-Domain Similarity Local Scaling (Conneau et al., 2018) because it is more resistant against hubness. Multiple iterations of Procrustes are run in total, with the intersections of forward and backward hypotheses from the previous round used as additional seeds. After five iterations, the hypothesis intersection is passed into the graph-matching algorithm.

After 20 iterations, H_f is returned as the final hypothesis and can be used to generate the dictionary.

4. Evaluation

The first segment outlines the evaluation methodology. In the second part, experiments are introduced, and their outcomes are discussed. We start with the main experiment of this thesis, BLI for Oshiwambo, and continue with a short verification of the implementation using a language pair, which was also covered in the experiments by Marchisio et al. (2022). We end the chapter with several experiments with Swahili, hoping to gain insight into some factors influencing the quality of the outcome. The code and evaluation results can be found on GitHub[10].

4.1. Methodology

As barely any available data can be used for an automated evaluation, it is done manually with the help of native Oshiwambo speakers who are also proficient in English. For 1800 words in the dictionary, the evaluators receive a ranked list of proposed translations and choose the best-ranked, correct translation (question type I). To combat human error, each word is evaluated at least twice. The final judgment will be made using a majority vote.

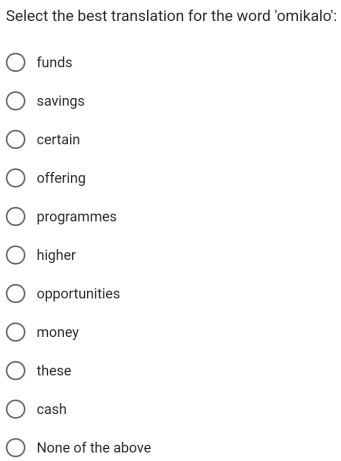

Figure 15: Sample survey question of type I choosing the best translation out of 10

Using those scores, we will calculate the $P@k$ score for $k \in \{1, 5, 10\}$ as it is the most commonly used metric in BLI. Nakashole (2019) has also used it, making my results comparable. Furthermore, we will also calculate the mean average precision (MAP) as recommended by Glavaš et al. (2019). The main advantage of MAP is that it rewards the model based on the ranking, whereas $P@k$ penalizes all correct translations at a rank $r > k$ equally (Laville et al., 2022). In our setting, where we only regard the first accurate translation, since it simplifies the evaluation, this is equivalent to computing

[10]https://github.com/addie9800/BA-BLI-for-Oshiwambo

the mean reciprocal rank (Glavaš et al., 2019), which is calculated using:

$$\text{MRR} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{rk}_i} \tag{65}$$

where N is the size of the test set and rk_i is the rank of the correct translation. If none of the translations is correct, choose $\frac{1}{\text{rk}_i} = 0$.

A problem that can arise when dealing with agglutinating languages is that words are not split correctly. For example, the word *kofikola*, which translates to *to school*, could be misinterpreted to mean *school*. Simply removing the *k* would correct the assignment in this case. To roughly estimate the magnitude of the issue, the survey contains a second question (type II) for half of the words, where the evaluators are asked to judge whether the top-ranked translation is "almost" a correct translation, where "almost" is explained using the previous example.

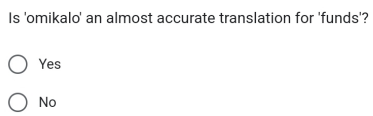

Figure 16: Sample survey question of type II determining "almost" correct translations

The URLs to the parts of the survey can be found in the appendix in section C.

4.2. BLI for Oshiwambo

The experiment's goal was to create a dictionary for the 3000 most frequent words in the English and Oshiwambo corpora, assuming that occurrences of a word and its translation are of a similar magnitude. The experiments were run as presented in section 3, using graph matching and iterative Procrustes. Most parameters relevant to the graph matching or Procrustes processes were used as in the original paper. To optimize the final dictionary, we performed a parameter search over several parameters:

- Vocabulary size or merge operations o in the training of the Byte Pair Embeddings $(10{,}000, 20{,}000, 50{,}000)$
- Computation of word embeddings using the mean or weighted average
- Extension of Oshiwambo corpus with a related language
- Number of seeds (50, 75, 100)
- Ending with Procrustes or graph matching

The experiment was run 10 times for each combination of parameters on a randomized seed input. The pseudorandom number generator was seeded for each iteration with the same value across all experiments. The evaluation was performed using the unused seeds, resulting in test set sizes of 164, 139, or 114, depending on the number of seeds specified. This is not ideal but determined by the limited resources at hand. The implementations of Marchisio et al. (2022) were used as a foundation upon which my experiments are set. The averaged P@1 scores of various parameter combinations range from $0.48\% - 3.54\%$. Unfortunately, these values are governed by a standard deviation of about 1, making it difficult to choose an ideal set of parameters based on these results.

Nevertheless, there are some observable tendencies. The clearest trait is that in our number of seeds regime, the best results are achieved when ending with Procrustes. A finding that coincides with the results of Marchisio et al. (2022). Also noticeable is out of the eight sets of parameters that averaged more than 3%, five had a vocabulary size of 20,000. The parameter combination that additionally used mean word embeddings and ended with Procrustes on the small corpus performed better than 3% for all numbers of seeds. The remaining parameters do not show any greater significance that could be confidently distinguished from fluctuating scores. The detailed results tables can be reviewed in the appendix in section E.

The parameters for the final dictionary were chosen in part based on the results of this grid search and on the results from Marchisio et al. (2022) if my results were inconclusive. Due to the unclear results, we chose multiple possible sets of parameters and evaluated them on the unused seeds. We selected the best-performing dictionary and used the 10 nearest neighbors for each word to generate the type I evaluation survey questions. We then randomly selected 50 % of the word pairs and generated the questions of type II.

Corpus Type	Word Embedding	Number of Seeds	Vocabulary Size	Ending with Procrustes	Best P@1
Small	**Weighted**	**100**	**20,000**	**True**	**7.83**
Extended	Mean	100	20,000	True	6.19
Small	Mean	100	20,000	True	5.26

Table 1: Best performing dictionaries for parameter selection

Table 2 presents the results of the manual evaluation. Responses were discarded if there was only one response to that question or the best answer was tied with the next best.

P@1	P@5	P@10	MRR	Valid Responses	Discarded Responses	Total Responses
1.65	4.35	6.86	2.97	1633	167	1800

Table 2: Scores of the Oshiwambo - English dictionary in manual evaluation - question type I

Comparing our $P@k$ scores for $k \in \{1, 5, 10\}$: 1.65, 4.35 and 6.86 with the scores of Nakashole (2019): 30, 56 and 58, we observe a significant decrease in performance.

We performed multiple follow-up experiments to locate possible issues, which will be presented in the following subsections.

Table 3 shows the results of evaluating the type II questions. Note that there is an annotation error, where a word was marked as correct but not "almost" correct. Also, 15 responses were discarded because no majority could be established, and 176 responses were discarded because the corresponding question of type I did not have a clear answer. The choice was made to remove responses to a type II question without a valid answer to the related type I question because "almost" correct and correct are indistinguishable in this case, providing no insight.

Type II \ Type I	Correct	Incorrect	Discarded
"Almost" Correct	10	4	-
Incorrect	1	782	-
Discarded	-	-	85 / 17

Table 3: Scores of the Oshiwambo - English dictionary in manual evaluation - question type II

The analysis of the results of the type II questions only shows a marginal increase in the P@1 score, with a change of +0.50%, if "almost" correct translations were to be considered correct. Furthermore, Table 4 shows that in the four cases where the best-ranked translation was only marked "almost" correct, the problem was not related to additional or missing syllables but rather words of similar meanings or from similar contexts.

Oshiwambo Word	"Almost" Correctly Translates to	Correct Translation
Omiyalu	Account	Amount
Aagundjuka	Young	Youth
Omuwiliki	Headed	*Director*
Evatelo	Providing	Assistance

Table 4: List of the four Oshiwambo words with the first translation choice marked as incorrect, but "almost" correct. The selected correct translation for the Oshiwambo word is additionally listed in the last column. The italicized translation was provided by a native speaker since it was not within the top 10 results

4.3. Implementation Verification

The experiment was repeated for English and German using the dictionaries from Marchisio et al. (2022) to rule out any significant errors in the implementation. The embeddings

were generated using the `Europarl` dataset (Tiedemann, 2012). Since the prior experiments did not show any significant influence on the choice of word embeddings, the test was run with mean embeddings. The system was set up to end with Procrustes and 100 seeds to make the results comparable to the Oshiwambo results. A parameter search was performed on the vocabulary size since this is a more language and corpus-specific parameter.

Vocabulary Size	Avg. Score	Best P@1
10,000	$\mathbf{52.08 \pm 2.34}$	**55.06**
20,000	51.24 ± 1.33	54.67
50,000	52.13 ± 0.69	52.90

Table 5: BLI for German - English

This is worse than the results achieved by Marchisio et al. (2022), who scored 59.4 in a mostly similar setup. A big difference between theirs and our experimental setup is the size of the final dictionary, which consists of 200,000 words in their case and 3000 in ours, making direct comparisons unfair to a certain degree. Yet, keeping this in mind, these scores are an improvement compared to their results in BLI using either Procrustes or graph matching, with the better result averaging 47.6. While it would be interesting to determine the origin of these performance differences, it is beyond the scope of this thesis. Nonetheless, these results show the basic functionality of the implementation.

4.4. Comparison with Swahili

There are two main questions this thesis is trying to answer. The first one, which Marchisio et al. (2022) have also formulated in their outlook, is how well (partly) graph-based approaches to BLI work on African languages, based on the example of Oshiwambo. The second one is how well it works in a low-resource setting, i.e., with small training corpora. Since experiments with Oshiwambo represent both challenges simultaneously, in the next step, the BLI is attempted for Swahili.

To analyze the disparity between the results of BLI for Oshiwambo and Swahili, another trial was run for Swahili with an artificially shrunken corpus to 40,000 sentences mimicking the amount of training data available for Oshiwambo. By reasoning similar to before, we limited the parameter search to the parameter o, the vocabulary size.

Corpus Type	Vocabulary Size	Avg. Score	Best P@1
Extended	10,000	37.30 ± 0.41	38.23
Extended	**20,000**	$\mathbf{39.18 \pm 0.97}$	**41.00**
Extended	50,000	37.39 ± 0.80	38.56
Small	10,000	23.79 ± 1.38	26.48
Small	20,000	25.07 ± 1.57	27.33
Small	50,000	25.40 ± 1.24	27.86

Table 6: BLI for Swahili with full-sized and reduced-sized corpora

In the best configuration, Swahili averages a $P@1$ score of 39.18. This is slightly worse than the lowest-performing translation pair examined by Marchisio et al. (2022). The English-Tamil pair scored 40.2 when ending with Procrustes for 100 seeds. The next best language pair, English-Bengali, averaged 45.3 in the same regime.

Limiting the size of the Swahili corpus has a noticeable effect and reduces the performance by $\sim 33\%$. Even though this is a significant decrease, this does not yet explain the even lower scores in Oshiwambo. One would expect similar scoring since Swahili and Oshiwambo are closely related. Another potential issue could be the disproportionality of the bilingual corpora. For Oshiwambo, the English corpus contains 544,353 sentences opposed by a 45,038 substantial Oshiwambo corpus. We accepted this inequality due to the inability to separate between parallel and extra content in the English corpus. In order to evaluate possible effects, we ran another experiment for Swahili, where the articles were required to be published in 2024, resulting in a 162,700 sentence English corpus (previously 380,089) and 42,049 (previously 130,052) line Swahili corpus.

Corpus Type	Vocabulary Size	Avg. Score	Best P@1
Small	**10,000**	$\mathbf{21.77 \pm 0.69}$	**22.91**
Small	20,000	21.14 ± 0.99	22.69
Small	50,000	18.19 ± 2.44	21.28

Table 7: BLI for Swahili with disproportionate corpora

Instead of raising the scores due to less irrelevant (not parallel) data in the English corpus, compared to the previous experiment, where only the Swahili corpus was truncated, the scores are lower. This raises the question of whether it is preferable to have more data, even if one language is overrepresented.

4.5. Isomorphism of Embedding Spaces

Another factor that helps to understand the differences in performance is the degree of isomorphism of the two embedding spaces. Using the implementations by Vulić et al. (2020), we calculate the Gromov-Hausdorff Distance and Laplacian Eigenvector similarities (EVS) for selected embedding spaces. Similar to the process of BLI, the

similarity measures are computed using the embeddings of the 3000 most frequent words from each language, which are calculated using mean word embeddings. The vocabulary size was chosen as 20,000.

Language Pair	Gromov-Hausdorff Distance	Eigenvector Similarity
English - Swahili	0.17	20.15
English - Oshiwambo	0.06	50.77
English - German	**0.15**	**4.58**

Table 8: Eigenvector similarity and Gromov-Hausdorff distances for selected language pairs

We received surprisingly good results for the different measures. As a reference, for the degree of isomorphy between English and German, Marchisio et al. (2022) measure 11.49 and 0.31 in Eigenvector similarity and Gormov-Hausdorff distance, respectively, as well as 56.66 and 0.26 for English-Tamil. Judging alone by this metric, Swahili-English should outperform Tamil, which is not the case. From a combinatorial perspective, the likely reason explaining this apparent contradiction is that less complex objects are generally more similar to others compared to more complex objects. The spaces studied in this thesis all contained 3000 data points, whereas the spaces studied by Marchisio et al. (2022) contained 200,000, making the values not comparable.

Nevertheless, they offer an indication of the three languages' varying performance. German, with very low values for both GH and EVS, is expected to perform noticeably better than Swahili and Oshiwambo. Even though Oshiwambo has a low GH distance, the dominating indicator seems to be the EVS. Figure 17 shows a regime of high values of EVS resulting in a strong drop in the performance of GOAT. With only three languages tested, it is impossible to name this as the issue confidently; tests with more languages are necessary, but it can be an explanation.

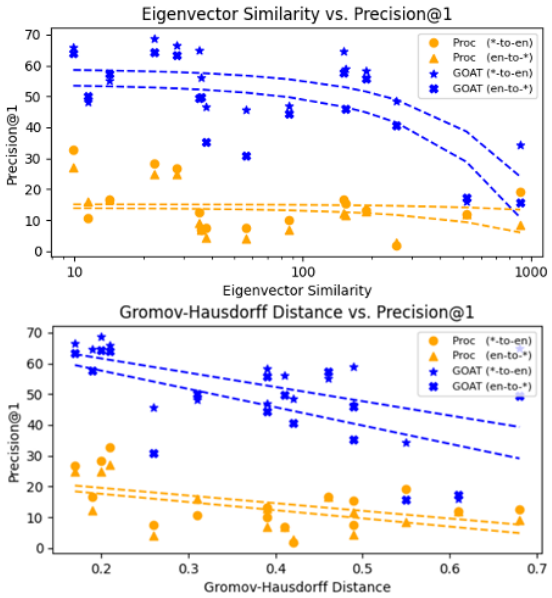

Figure 17: Dependency graphs of P@1 scores to isomorphy metrics. Reproduced under CC 4.0 BY from Marchisio et al. (2022)

4.6. Choice of Embeddings

Finally, we want to evaluate the choice of subword-based embeddings. The experiments are repeated using conventional `GloVe` embeddings (Pennington et al., 2014) trained on the original corpora for each language pair.

Language Pair	Avg. Score	Best P@1	Gromov-Hausdorff Distance	Eigenvector Similarity
English-Oshiwambo	1.24 ± 0.64	1.90	0.11	19.57
English-Swahili	$\mathbf{45.10 \pm 0.85}$	**46.67**	**0.08**	**8.10**

Table 9: Performance and isomorphism measures of word embeddings

We observe a 7.5 average point increase for Swahili and a 2.3 average point decrease for Oshiwambo when attempting BLI with word embeddings. This disparity allows for no definite answer to the optimal embedding question. Interestingly, though, the measures for isomorphy in the English-Oshiwambo experiment are very similar to the previous experiment for Swahili-English using Byte Pair embeddings (Sennrich et al., 2016). Yet, the qualities of the generated dictionaries diverge. Within the setting of word embeddings, the measures are consistent with previous observations of a correlation between them and the performance in BLI. It raises the question of how comparable these measures of isomorphy are across varying experimental setups.

5. Discussion

I get very poor results in BLI for Oshiwambo, with the best dictionary having a $P@1$ score of 1.65. The poor performance can also not be explained by issues due to the agglutinative nature of Oshiwambo. When switching to Swahili, the results are significantly better, almost as good as those for distant language pairs in the original paper by Marchisio et al. (2022). My scores are not easily comparable with theirs because of the larger training corpus and larger dictionary size in the original experiments. Still, they do provide an adequate point of reference.

The results for Swahili show that BLI can be successful in the Bantu languages. It remains unclear why the difference in Swahili and Oshiwambos' performance is so significant. An indicator is that with the current training corpora, the embeddings generated for English-Oshiwambo are measurable *less isomorphic* than those generated for English-Swahili. The claim that this influences the scoring so significantly would need to be verified by further experiments with, ideally, other related languages in a similar training setup. A direct comparison of the values with the results from Marchisio et al. (2022) again does not make sense due to the different embedding space sizes. Such a comparison would lead to a much higher expected performance for all examined language pairs. Further efforts can be invested into determining factors influencing the isomorphy scores and their influence on BLI.

I also performed multiple tests with varying corpora. I observed that the low amount of available training data in Oshiwambo is likely partly responsible for its poor performance. Still, the lack of training data cannot fully explain it. Swahili still performed 18.22 percentage points better than Oshiwambo when trained on a similar number of sentences. Extending the Oshiwambo corpus with a related, higher-resource language showed no notable differences. Reducing the mismatch in corpus size seemed to influence the overall results negatively, most likely caused by insufficient training data for generating high-quality embeddings. A question that arises in this context is how the performance scales with more training data. With only 130,052 sentences, the embeddings were trained on a significantly smaller dataset compared to the `fastText` (Bojanowski et al., 2017) embeddings trained on Wikipedia dumps, which Marchisio et al. (2022) used.

Finally, no definite observation can be made about the effect of subword embeddings: Swahili scored better with word embeddings, whereas Oshiwambo scored better using subword embeddings. Hence, the effectiveness of subword embeddings should be further investigated. They intuitively should have an advantage over word embeddings for agglutinative languages.

During the evaluation, it became clear that in Oshiwambo, due to its limited vocabulary, the meanings of words are partially context-dependent (T. Ndahutuka, personal communication, 19.02.2025). This suggests that further research with contextual word embeddings potentially improves performance.

To conclude, one must recognize that all findings are primarily done in one language or by comparing two languages. To present more robust statements, the experiments must be repeated with more languages in a similar setting, preferably related languages, to rule out possible language-specific issues as best as possible.

References

Yonathan Aflalo, Alexander Bronstein, and Ron Kimmel. 2015. On convex relaxation of graph isomorphism. *Proceedings of the National Academy of Sciences*, 112(10):2942–2947.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2019. Bilingual lexicon induction through unsupervised machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5002–5007, Florence, Italy. Association for Computational Linguistics.

Atlas of Namibia Team. 2022. *Atlas of Namibia: its land, water and life*. Namibia Nature Foundation, Windhoek.

R. Balakrishnan and S. Sridharan. 2018. *Foundations of Discrete Mathematics with Algorithms and Programming*, 1st edition. Chapman and Hall/CRC.

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

Siegfried Bosch. 2014. *Lineare Algebra*, 5 edition. Springer-Lehrbuch. Springer Spektrum Berlin, Heidelberg.

Lindell Bromham, Russell Dinnage, Hedvig Skirgård, Andrew Ritchie, Marcel Cardillo, Felicity Meakins, Simon Greenhill, and Xia Hua. 2022. Global predictors of language endangerment and the future of linguistic diversity. *Nature Ecology & Evolution*, 6.

Rainer Burkard, Mauro Dell'Amico, and Silvano Martello. 2012. *Assignment Problems*. Society for Industrial and Applied Mathematics.

Sanghyuk Choi, Taeuk Kim, Jinseok Seol, and Sang-goo Lee. 2017. A syllable-based technique for word embeddings of Korean words. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 36–40, Copenhagen, Denmark. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data.

Thera Marie Crane, Karl Lindgren-Streicher, and Andy Wingo. 2004. *Hai ti! A beginner's guide to Oshikwanyama*. US Peace Corps, Namibia, Namibia.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transportation distances.

Max Dallabetta, Conrad Dobberstein, Adrian Breiding, and Alan Akbik. 2024. Fundus: A simple-to-use news scraper optimized for high quality extractions.

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

Qiuyu Ding, Hailong Cao, Zihao Feng, and Tiejun Zhao. 2024. Enhancing isomorphism between word embedding spaces for distant languages bilingual lexicon induction. *Neural Computing and Applications*. Cited by: 0.

Georgiana Dinu and Marco Baroni. 2014. Improving zero-shot learning by mitigating the hubness problem. *CoRR*, abs/1412.6568.

Wei Dong, Charikar Moses, and Kai Li. 2011. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, page 577–586, New York, NY, USA. Association for Computing Machinery.

David M. Eberhard, Gary F. Simons, and Charles D. Fennig, editors. 2024. *Ethnologue: Languages of the World*, twenty-seventh edition. SIL International, Dallas, Texas. Online version.

ELCIN Church Council Special Committees. 1996. *English - Ndonga Dictionary*. ELCIN Printing Press, Ondangwa.

Donniell E. Fishkind, Sancar Adali, Heather G. Patsolic, Lingyao Meng, Digvijay Singh, Vince Lyzinski, and Carey E. Priebe. 2019. Seeded graph matching. *Pattern Recognition*, 87:203–215.

M. Frank and P. Wolfe. 1956. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38.

Goran Glavaš, Robert Litschko, Sebastian Ruder, and Ivan Vulić. 2019. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 710–721, Florence, Italy. Association for Computational Linguistics.

Google Ireland Limited. Google translate. Accessed: 2024-08-12.

Zellig S. Harris. 1954. Distributional structure. *WORD*, 10(2-3):146–162.

Wendi A. Haugh. 2022. Culture summary: Ovambo. Accessed: 2024-07-02.

Benjamin Heinzerling and Michael Strube. 2018. BPEmb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

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

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6282–6293, Online. Association for Computational Linguistics.

Paul Knopp and Richard Sinkhorn. 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343 – 348.

Tjalling C. Koopmans and Martin Beckmann. 1957. Assignment problems and the location of economic activities. *Econometrica*, 25(1):53–76.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

H. W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Martin Laville, Emmanuel Morin, and Phillippe Langlais. 2022. About evaluating bilingual lexicon induction. In *Proceedings of the BUCC Workshop within LREC 2022*, pages 8–14, Marseille, France. European Language Resources Association.

Nada Lavrač, Vid Podpečan, and Marko Robnik-Šikonja. 2021. *Representation Learning: Propositionalization and Embeddings*. Springer.

Yaoyiran Li, Anna Korhonen, and Ivan Vulić. 2023. On bilingual lexicon induction with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9577–9599, Singapore. Association for Computational Linguistics.

Kelly Marchisio, Youngser Park, Ali Saad-Eldin, Anton Alyakin, Kevin Duh, Carey Priebe, and Philipp Koehn. 2021. An analysis of Euclidean vs. graph-based framing for bilingual lexicon induction from word embedding spaces. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 738–749, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kelly Marchisio, Ali Saad-Eldin, Kevin Duh, Carey Priebe, and Philipp Koehn. 2022. Bilingual lexicon induction for low-resource languages using graph matching via optimal transport. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2545–2561, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Adolfo C. Mascarenhas. 2025. Zanzibar and pemba. Accessed: 2025-01-06.

Sunil Mathew, John N. Mordeson, and M. Binu. 2023. *Weighted and Fuzzy Graph Theory*, 1 edition. Studies in Fuzziness and Soft Computing. Springer Cham.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality.

Ministry of Home Affairs, Immigration, Safety & Security. The constitution of the republic of namibia. Accessed: 2024-08-15.

Facundo Mémoli. 2008. Gromov-hausdorff distances in euclidean spaces. pages 1–8.

Ndapa Nakashole. 2019. Bilingual dictionary induction for bantu languages.

Namibia Statistics Agency (NSA). 2013. *Namibia Population and Housing Census 2011*. NSA, Windhoek.

Wilhelmina Nekoto, Julia Kreutzer, Jenalea Rajab, Millicent Ochieng, and Jade Abbott. 2022. Participatory translations of oshiwambo: Towards sustainable culture preservation with language technology. In *3rd Workshop on African Natural Language Processing*.

Aitor Ormazabal, Mikel Artetxe, Gorka Labaka, Aitor Soroa, and Eneko Agirre. 2019. Analyzing the limitations of cross-lingual word embedding mappings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4990–4995, Florence, Italy. Association for Computational Linguistics.

Oshikwanyama Curriculum Committee. 2019. *Oshikwanyama-English/EnglishOshikwanyama Dictionary*. Namibia Publishing House : Macmillan Education Namibia, Windhoek, Namibia.

Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R Gormley, and Graham Neubig. 2019. BLISS in non-isometric embedding spaces.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović;. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(86):2487–2531.

Ali Saad-Eldin, Benjamin D. Pedigo, Carey E. Priebe, and Joshua T. Vogelstein. 2021. Graph matching via optimal transport.

Sonal Sannigrahi and Jesse Read. 2022. Isomorphic cross-lingual embeddings for low-resource languages.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Casper S. Shikali, Zhou Sijie, Liu Qihe, and Refuoe Mokhosi. 2019. Better word representation vectors using syllabic alphabet: A case study of swahili. *Applied Sciences*, 9(18).

Edward Shikesho. 2019. A critical evaluation of english-kwanyama dictionary by g.w.r. tobias and b.h.c. turvey.

Anders Søgaard, Sebastian Ruder, and Ivan Vulić. 2018. On the limitations of unsupervised bilingual dictionary induction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 778–788, Melbourne, Australia. Association for Computational Linguistics.

Justin Solomon. 2017. Computational optimal transport.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

T.E. Tirronen. 1986. *Ndonga-English Dictionary*. ELCIN.

G. W Tobias. 1954. *English-Kwanyama Dictionary / G. W. R. Tobias and B. H. C. Turvey*, 1. ed. edition. Johannesburg.

B.H.C. Turvey, W. Zimmermann, and G.B. Taapopi. 1977. *Kwanyama-English Dictionary*. Witwatersrand University Press.

Hermann Tönjes. 1910a. *Lehrbuch der Ovambo-Sprache, Osikuanjama*. De Gruyter Mouton, Berlin, Boston.

Hermann Tönjes. 1910b. *Wörterbuch der Ovambo-Sprache : Osikuanjama - Deutsch / von Hermann Tönjes*. Lehrbücher des Seminars für Orientalische Sprachen zu Berlin BV000896676 25. Berlin.

Upgrading African Languages Project Namibia. 2004. *Oshikwanyama Omushangelo 3: Oshikwanyama Orthography 3*. Gamsberg Macmillan.

Johannes Jurgens Viljoen, Petrus Amakali, and Martha Namuandi. 1984. *Oshindonga/English - English/Oshindonga Dictionary*. Macmillan Education Namibia Publishers (Pty) Ltd., Namibia.

Joshua T. Vogelstein, James M. Conroy, Vince Lyzinski, Lauren J. Podrazik, Sean G. Kratzer, Eric T. Harley, Donniell E. Fishkind, Randal J. Vogelstein, and Carey E. Priebe. 2015. Fast approximate quadratic programming for graph matching. *PLoS One*, 10(4):e0121002.

John von Neumann. 1953. *1. A Certain Zero-sum Two-person Game Equivalent to the Optimal Assignment Problem*, pages 5–12. Princeton University Press, Princeton.

Erika von Wietersheim. 2018. *The Constitution of the Republic of Namibia: Annotated Edition*. Konrad-Adenauer-Stiftung, Windhoek, Namibia.

Erika von Wietersheim and Rauna N. Ndinoshiho. 2017. *Efinamhango loRepublika yaNamibia*. Konrad-Adenauer-Stiftung and Namibian Scientific Society, Windhoek, Namibia.

Ivan Vulić, Sebastian Ruder, and Anders Søgaard. 2020. Are all good word vector spaces isomorphic? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Benji Wald. 2003. *Swahili and the Bantu Languages*, pages 991–1014. Routledge.

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

A. Derivation of the Closed-Form Solution the Procrustes Problem

The closed-form solution was originally derived by Schönemann (1966). To find it, we start by expressing the equation

$$\underset{W \in O(d)}{\arg\min} \|XW - Z\|_F^2 \tag{66}$$

using the trace formulation:

$$g_1 := \operatorname{tr}\left(W^\top X^\top X W - 2W^\top X^\top Z + Z^\top Z\right) \tag{67}$$

The orthogonality condition can be represented using a matrix $L \in \mathbb{R}^{d \times d}$ of Lagrange Multipliers:

$$g_2 := \operatorname{tr}\left(L(W^\top W - \mathbb{1}_n)\right) \tag{68}$$

For $g = g_1 + g_2$, the partial derivative with respect to W, is given by:

$$\frac{\partial g}{\partial W} = (\underbrace{X^\top X}_{P} + X^\top X)W - 2\underbrace{X^\top Z}_{S} + W\underbrace{(L + L^\top)}_{\frac{Q}{2}} \tag{69}$$

To find an extremum of g_1, the derivative needs to be equal to 0, i.e.

$$S = PW + WQ \tag{70}$$

Since P, Q are symmetric matrices, we get from equation 70:

$$Q = W^\top S - W^\top P W = Q^\top \implies W^\top S = S^\top W \tag{71}$$

This allows us to extract the following relationship:

$$SS^\top = TS^\top ST^\top \tag{72}$$

which leaves us with two symmetric matrices $S^\top S, SS^\top$. These can be diagonalized using orthonormal matrices and share the same Eigenvalues. Hence, for orthonormal $U, V \in \mathbb{R}^{d \times d}$:

$$SS^\top = UDU^\top \qquad S^\top S = VDV^\top \tag{73}$$

This gives us what we wanted to show:

$$UDU^\top = WVDV^\top W^\top \implies W = UV^\top \tag{74}$$

One can easily convince oneself, that for a given X, Z, U, V can be computed using the Singular Value Decomposition $S = UD^{\frac{1}{2}}V^\top$ of S. If we apply this to our instance of the Procrustes Problem, we receive a closed-form solution (Conneau et al., 2018):

$$W = \underset{W \in O(d)}{\arg\min} \|WX - Z\|_F^2 = UV^\top \qquad UD^{\frac{1}{2}}V^\top = \text{SVD}(ZX^\top) \tag{75}$$

B. List of Name Seeds

namibia, namibian, south, national, million, african, windhoek, swapo, economic, international, group, united, chief, human, china, union, action, democratic, centre, zimbabwe, june, west, organisation, democracy, france, authority, progress, association, hydrogen, freedom, pohamba, movement, botswana, park, congress, geingob, nujoma, john, zuma, rally, sadc, india, oshakati, mugabe, mbumba, zambia, standard, unam, primary, johannesburg, michael, david, angula, russia, katutura, kavango, nandi-ndaitwah, keetmanshoop, erongo, khomas, robert, peter, george, hage, corporation, spyl, ohangwena, rundu, venaani, mozambique, martin, tsvangirai, nangolo, beukes, transnamib, omusati, hifikepunye, shikongo, combined, caprivi, oshana, ondangwa, moses, mutual, hardap, kunene, mbeki, joseph, johannes, thomas, sudan, swartbooi, ongwediva, karas, simon, mulilo, nampower, richard, ekandjo, hiv-aids, uganda, alweendo, rwanda, netumbo, harare, namwater, paulus, nudo, tobias, shiimi, namfisa, amushelelo, namcor, hamutenya, morgan, petrus

C. List of Survey URLs

The 8 surveys can be found at these URLs:

1. https://forms.gle/P718jQgn6AEK2dVAA
2. https://forms.gle/u6CuEFyNeYmz1hBa8
3. https://forms.gle/R3UMF49tGcehThin7
4. https://forms.gle/mGBdZ4RUCJxxFeRz9
5. https://forms.gle/PWgXpk3Z8R7hZwcQ8
6. https://forms.gle/Ud5CWGrykxzAckGL6
7. https://forms.gle/6jDFqUUHH97vg9b38
8. https://forms.gle/ZG2vaFmeJCbvQd3GA

D. Visualization of Step 2 of The Hungarian Algorithm

Symbol	Use in Diagram
i	Index of rows of Q
j	Index of columns of Q
k	Tally of length of sequence of 1's and 1*'s & Tally to clear essential rows
l	Tally to test distinctness of i_{k+1} from $i_1, \cdots, i_k$
$i_1, i_2, \cdots, i_n$	Record of rows in sequence of 1's and 1*'s
$j_0, j_1, \cdots, j_{n-1}$	Record of columns in sequence of 1's and 1*'s
$\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_n$	Record of essential rows

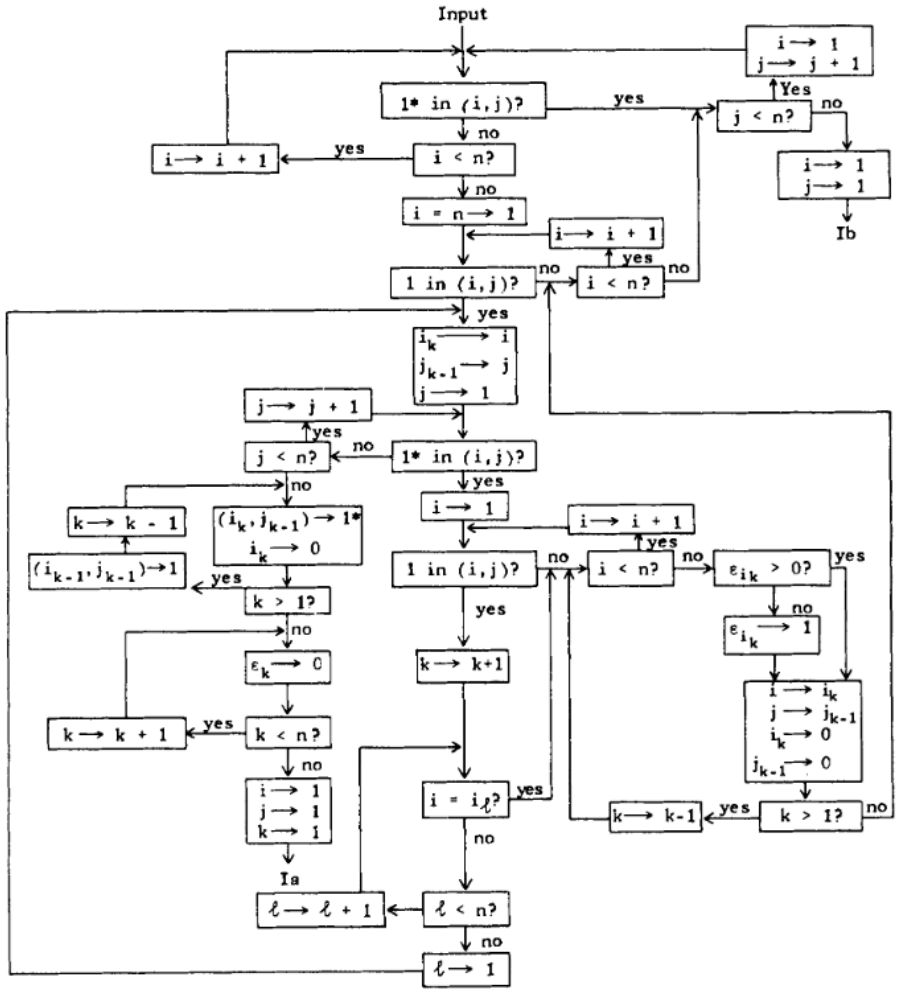

Figure 18: Visualization of Step 2 of the Hungarian Algorithm (Kuhn, 1955)

E. Parameter Search Results

Embedding Computation	Number of Seeds	Vocabulary Size	End with Procrustes	Averaged Score
Mean	50	10,000	True	0.85 ± 0.82
Mean	50	10,000	False	1.16 ± 0.84
Mean	75	10,000	True	0.93 ± 0.89
Mean	75	10,000	False	1.50 ± 0.71
Mean	100	10,000	True	0.71 ± 0.70
Mean	100	10,000	False	1.50 ± 1.56
Mean	**50**	**20,000**	**True**	$\mathbf{3.05 \pm 1.11}$
Mean	50	20,000	False	1.66 ± 0.96
Mean	**75**	**20,000**	**True**	$\mathbf{3.19 \pm 1.54}$
Mean	75	20,000	False	1.57 ± 0.81
Mean	**100**	**20,000**	**True**	$\mathbf{3.54 \pm 1.02}$
Mean	100	20,000	False	2.32 ± 0.96
Mean	50	50,000	True	2.00 ± 0.81
Mean	50	50,000	False	1.21 ± 0.57
Mean	75	50,000	True	2.45 ± 1.486
Mean	75	50,000	False	1.96 ± 0.91
Mean	100	50,000	True	2.22 ± 1.57
Mean	100	50,000	False	2.19 ± 0.955
Weighted	50	10,000	True	1.77 ± 1.36
Weighted	50	10,000	False	1.68 ± 1.22
Weighted	75	10,000	True	1.96 ± 1.45
Weighted	75	10,000	False	1.45 ± 0.59
Weighted	100	10,000	True	1.83 ± 0.86
Weighted	100	10,000	False	1.64 ± 1.25
Weighted	50	20,000	True	2.28 ± 1.05
Weighted	50	20,000	False	1.77 ± 0.73
Weighted	75	20,000	True	2.59 ± 0.97
Weighted	75	20,000	False	1.96 ± 0.97
Weighted	100	20,000	True	2.96 ± 1.24
Weighted	100	20,000	False	2.26 ± 0.73
Weighted	50	50,000	True	2.52 ± 1.14
Weighted	50	50,000	False	1.84 ± 1.19
Weighted	**75**	**50,000**	**True**	$\mathbf{3.07 \pm 1.47}$
Weighted	75	50,000	False	1.21 ± 0.48
Weighted	100	50,000	True	1.91 ± 1.63
Weighted	100	50,000	False	1.45 ± 1.28

Table 10: Parameter search results using the Oshiwambo corpus, extended with the Swahili corpus and averaged over 10 iterations

Embedding Computation	Number of Seeds	Vocabulary Size	End with Procrustes	Averaged Score
Mean	50	10,000	True	2.23 ± 1.39
Mean	50	10,000	False	1.39 ± 0.86
Mean	**75**	**10,000**	**True**	$\mathbf{3.17 \pm 1.28}$
Mean	75	10,000	False	1.15 ± 0.61
Mean	100	10,000	True	2.57 ± 1.21
Mean	100	10,000	False	1.58 ± 1.00
Mean	50	20,000	True	2.48 ± 1.23
Mean	50	20,000	False	1.53 ± 0.83
Mean	75	20,000	True	2.09 ± 0.72
Mean	75	20,000	False	1.06 ± 1.07
Mean	100	20,000	True	2.54 ± 1.73
Mean	100	20,000	False	1.74 ± 0.92
Mean	50	50,000	True	1.39 ± 0.70
Mean	50	50,000	False	0.72 ± 0.47
Mean	75	50,000	True	2.39 ± 0.97
Mean	75	50,000	False	1.08 ± 0.61
Mean	100	50,000	True	2.32 ± 1.05
Mean	100	50,000	False	2.30 ± 1.97
Weighted	50	10,000	True	2.68 ± 1.23
Weighted	50	10,000	False	1.23 ± 0.58
Weighted	**75**	**10,000**	**True**	$\mathbf{3.09 \pm 1.08}$
Weighted	75	10,000	False	1.45 ± 0.59
Weighted	100	10,000	True	2.84 ± 1.22
Weighted	100	10,000	False	1.70 ± 0.89
Weighted	50	20,000	True	2.16 ± 1.31
Weighted	50	20,000	False	0.67 ± 0.73
Weighted	**75**	**20,000**	**True**	$\mathbf{3.26 \pm 1.01}$
Weighted	75	20,000	False	1.57 ± 1.25
Weighted	**100**	**20,000**	**True**	$\mathbf{3.01 \pm 0.95}$
Weighted	100	20,000	False	2.00 ± 0.82
Weighted	50	50,000	True	2.30 ± 0.80
Weighted	50	50,000	False	0.48 ± 0.74
Weighted	75	50,000	True	2.21 ± 0.98
Weighted	75	50,000	False	1.14 ± 0.84
Weighted	100	50,000	True	2.65 ± 0.93
Weighted	100	50,000	False	1.06 ± 0.37

Table 11: Parameter search results using the Oshiwambo corpus averaged over 10 iterations

Erklärung über den Einsatz von generativer KI

In der Erstellung dieser Studienarbeit wurde punktuell auf den Einsatz von generativer Künstlicher Intelligenz (KI) zurückgegriffen. Generative KI wurde für vier Zwecke verwendet:

1. Rohversionen einzelner Abbildungen wurden unter der Verwendung von `ChatGPT` erstellt. Betroffene Abbildungen sind mit dem Vermerk [GPT] gekennzeichnet.

2. Einzelne Bestandteile des Programmcodes wurde unter der Verwendung von `ChatGPT` erstellt. Betroffene Funktionen und Dateien sind mit einem entsprechenden Vermerk gekennzeichnet.

3. Bei der Erstellung des Programmcodes wurde des Weiteren `GitHub Copilot` zur Codevervollständigung genutzt.

4. Mit dem Tool `Grammarly` wurde diese Arbeit auf orthografische Fehler überprüft und einzelne Formulierungen angepasst.

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen, Bilder sowie die Nutzung von Künstlicher Intelligenz für die Erstellung von Texten und Abbildungen, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den February 25, 2025 .....