Injecting Explicit Cross-lingual Embeddings into Pre-trained Multilingual Models for Code-Switching Detection

Thapelo Sindane[1], Vukosi Marivate[1 2], and Avashlin Moodley

[1] Data Science For Social Impact Lab, University of Pretoria, Pretoria,
sindane.thapelo@tuks.co.za,
[2] Lelapa AI,
vukosi.marivate@up.ac.za
[3] The Council of Scientific and Industrial Research,
amoodley1@csir.co.za

Abstract. Code-switching has become the modus operandi of internet communication in many communities, such as South Africans, who are domestically multilingual. This phenomenon has made processing textual data increasingly complex due to non-standard ways of writing, spontaneous word replacements, and other challenges. Pre-trained multilingual models have shown elevated text processing capabilities in various similar downstream tasks such as language identification, dialect detection, and language family discrimination. In this study, we extensively investigate the use of pre-trained multilingual models - AfroXLMR, and Serengeti for code-switching detection on five South African languages: Sesotho, Setswana, IsiZulu, IsiXhosa, and English, with English used interchangeably with the other four languages, including various transfer learning settings. Additionally, we explore the modeling of known switching pairs within a dataset through explicit cross-lingual embeddings extracted using projection models: VecMap, Muse, and Canonical Correlation Analyses (CCA). The resulting cross-lingual embeddings are used to replace the embedding layer of a pre-trained multilingual model without additional training. Concretely, our results show that performance gains can be realized (from 59.1% monolingual to 74.1% cross-lingual, and to 90.8% multi-lingual) by closing the representational gap between the languages of the code-switched dataset with known codes, using cross-lingual representations. Moreover, expanding code-switched datasets with datasets of closely related languages improves code-switching classification, especially in cases with minimal training examples.

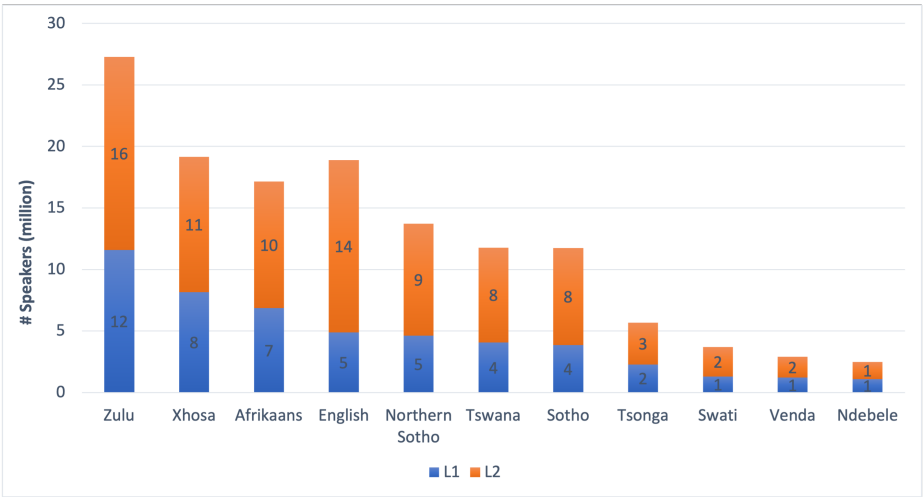
Keywords: Pre-trained Multilingual Models, Cross-lingual Embeddings, Code-switching Detection

1 Introduction

Code-switching, a term used interchangeably with code-mixing in Natural Language Processing (NLP), refers to a linguistic phenomenon where a single utterance or text is made up of multilingual tokens or words (referred to as codes), arranged meaningfully to

the receiving audience, prevalent in multilingual communities such as the Internet [18]. In a typical South African household setting, the majority of families are multilingual, as seen in Figure 1[4]. This figure highlights the number of people using each language (e.g, Zulu) as the first or preferred language, and alternatively, how many people use it as a second additional language. However, language technologies developed for these South African low-resourced languages are monolingual in nature. This myopic view limits the ability of these users to fully express themselves within these technologies, thus motivating a need for modeling code-switching in speech or text understanding technologies.

Fig. 1: Number of First Additional (L1), and Second additional Speakers (L2) in a South African Household.



Code-switching is not a new challenge. In 2013, [24] investigated the implications of code-switching for developing automated speech recognition systems using Sepedi and English code-switched datasets sourced from radio broadcasts. In [37], the authors investigated code-switching detection formalized as language detection together with Part-of-speech (POS) tagging. While [33] explored the creation of POS datasets from code-switched tweets that closely resemble real-world scenarios. On the other hand, [3] proposed a shared task for modeling Named Entity Recognition (NER) on code-switched datasets. Closer to our methodology, [17] proposed a hybrid model composed of an attention mechanism and a recurrent neural network for code-switching detection.
However, none of the aforementioned works attempted a natural standpoint that brings the involved languages in the code-switched dataset closer together through cross-lingual representation learning. Cross-lingual representation learning emerges from a

[4] https://en.wikipedia.org/wiki/Languages_of_South_Africa

broader field of cross-lingual models, where the idea is to fuse two or more syntactic and semantically sound monolingual embeddings in order to supplement the linguistic shortcomings of both embedding spaces for improving downstream task performance [23, 15, 5]. We hypothesize that given known languages $L_g = \{l1, l2, ..., ln\}$ in the labeled code-switched dataset C_{cs}, developing a language model M_L through cross-lingual representation learning of languages in L_g can improve on the task of code-switching classification (and possibly generalize to other extensions of code-switching tasks such as code-switched POS, code-switched NER, and more complex tasks like code-switched question answering) following the linguistic equivalence constraint of code-switching. Using cross-lingual representation learning to bring the languages in the code-switched datasets closer together may establish a representational space that closely matches code-switching, thus improving training, contrary to treating the codes as separate entities during training. Subsequently, we then want to answer the question: what effect do explicit cross-lingual representations generated from the involved language have on the performance of code-switching classification? To answer this, we conduct a systematic analysis and evaluation of supervised models using both monolingual and explicit cross-lingual embeddings on the task of word-level code-switching classification. The field of cross-lingual models has witnessed great success over the years in NLP and has evolved to advanced processing capabilities using large pre-trained multilingual models, where cross-lingual representation is implicitly learned [12, 10]. On this, our methodology investigates the downstream performance of code-switching classification prior to the injection of cross-lingual embeddings into the embedding layer of transformer architectures, AfroXLM-r, and Serengeti as our training and classification models. To the best of our knowledge, this work is the first study to investigate the injection of cross-lingual embeddings into pretrained multi-lingual models for code-switching detection or classification, regardless of the outcomes.

In summary, our major contributions are succinctly organized as follows:

1. We investigate the use of cross-lingual embeddings in the context of code-switching classification for four South African low-resource languages, namely, Setswana, Sesotho, isiZulu, and IsiXhosa, and one high-resourced language - English.
2. We showcase the experimental results that highlight an intuitive relationship between code-switching and cross-lingual representations, through linear improvements of code-switching detection from monolingual embeddings, to explicit cross-lingual embeddings, all the way to implicit multilingual representational training.
3. We highlight characteristics that are deterministic of improved performance for code-switching classification using cross-lingual and monolingual embeddings.

The rest of the paper is organized as follows: Section 2 discusses the related works, Section 3 outlines the methodology, broken down into the discussion of the monolingual datasets for training embeddings, an outline of our multilingual lexicons, a description of our code-switching datasets, details on pre-processing steps, techniques for generating monolingual embeddings, techniques for generating cross-lingual embeddings, details of pre-trained models used, the approach for injecting embeddings into pre-trained models, and the experimental setup for the aforementioned sub-section. Section 4 presents the experimental findings and results of this study, while Section 5 provides analyses of the findings. Finally, Section 6 gives the concluding remarks.

2 Related Work

Recent studies show the need to accelerate code-switching inclusion in language technology developments and the limitations incurred if otherwise [34]. In line with this shared responsibility, various works addressing different challenges across different downstream tasks have been proposed for code-switching. For code-switching detection, an early work by [9] proposed a Conditional Random Field (CRF) model based on their success in sequence labeling and a sub-word segmental approach that segments lexical units according to their morphological grounding. [40] explores sub-word vectors from [8], prefix and suffix extraction, and linear-chain CRF for code-switching detection and reports (at the time) promising results (83.0 $\%$, 94.9 $\%$ F1 score, and accuracy respectively). While [27] remedies incorrect annotations of borrowed words in the code-switched text by proposing a set of likelihood metrics that utilize language usage patterns on Twitter.

Code-switching classification as a stand-alone downstream task has use-case limitations, and as such, various works have coupled it with other downstream tasks. [35] exploits existing probabilistic tree-based taggers to generate training features of the switched languages for POS tagging in the context of code-switching. They use heuristic-driven methods to combine these information-rich features to assist machine-learning-based prediction. [7] uses a joint Factorial CRF model to process complex trilingual code-mixed data for code-switching detection and POS tagging simultaneously. In contrast, [16] proposes code-switching-tailored lexical normalizers to improve the impact of non-canonical data points on POS tagging.

For code-switching in the context of NER, [6] takes an architecture-driven approach and proposes a hybrid model consisting of enriched pre-trained embeddings, a Bidirectional Long and Short Term Memory (Bi-LSTM) model to process the left and right context over the continuous representations, a convolution layer to model spatial dependencies, and finally a CRF for sequence prediction. Closer to our work, [39] concatenates English and Spanish monolingual pre-trained embeddings together with character-level representation to address the out-of-vocabulary (OOV) issue. In contrast to our work, they did not explore cross-lingual representations. In the advent of the attention era, [38] addresses both NER and code-switching detection using multilingual meta-embeddings extracted using a fully connected neural network and an attention mechanism layer.

In the aforementioned works, South African low-resourced languages lag behind across multiple tasks and techniques for processing code-switching text. In an attempt to remedy this limitation as well as ignite research interest, we propose a systematic analysis of using both monolingual and cross-lingual vector representations for code-switching detection, together with injecting static cross-lingual embeddings into the embeddings layer of the transformer architectures. In this study, we were able to show a progressive improvement over monolingual embeddings training to explicit cross-lingual embedding training, and thus implicit multilingual (i.e. of higher cross-lingual quality) embedding training, highlighting an intuitive relationship between languages involved in shared representation space, and the modeling of code-switching.

3 Methodology

3.1 Corpora

This study explores four low-resourced South African languages (LRSAL), namely, Sesotho, Setswana, isiZulu, and IsiXhosa, and one high-resourced language - English with codes: sot, tsn, zul, xho, and eng, respectively. These languages are chosen based on the availability of code-switching datasets. The sources for our monolingual corpora are: Flores [11], Conference on Machine Translation (WMT) [11], multilingual colossal clean crawled corpus (MC4) [30], National Center for Language Human Technology (NCHLT) [14], and African Crawl Dataset [36]. The statistics of our monolingual data are outlined in Table 1. This table contains the merged monolingual datasets for the four languages, where the columns 'sentences', and 'Unique vocabulary' indicate the number of sentences and unique words after the merge. The columns 'After LID', and 'Unique vocabulary' outline the number of sentences and unique vocabulary that remained after performing language identification using GlotLID [19] as mentioned under pre-processing below.

Table 1: This table outlines the monolingual datasets collected sources: Flores [11], WMT [11], MC4 [30], and NCHLT [14].

Lang.	Sentences	Unique Voc	After LID	Unique Voc
tsn	1.1M	388K	462K	118K
sot	2.6M	1.5M	750K	453K
xho	2.7M	2.8M	1.2M	1.2M
zul	7.6M	9.2M	1.5M	1.5M

3.2 Bilingual Lexicons

We used bilingual lexicons to generate the cross-lingual representation from the following sources: Cape Peninsula University of Technology (CPUT)[5], Open Education Resource Term Bank (OERTB) [29], and our manually collected lexicons from government public school repositories[6]. The collection resulted in 8742 en-tsn, 8763 en-sot, 11117 en-xho, and 17406 en-zul bilingual lexicon pairs.

3.3 Code-Switched dataset

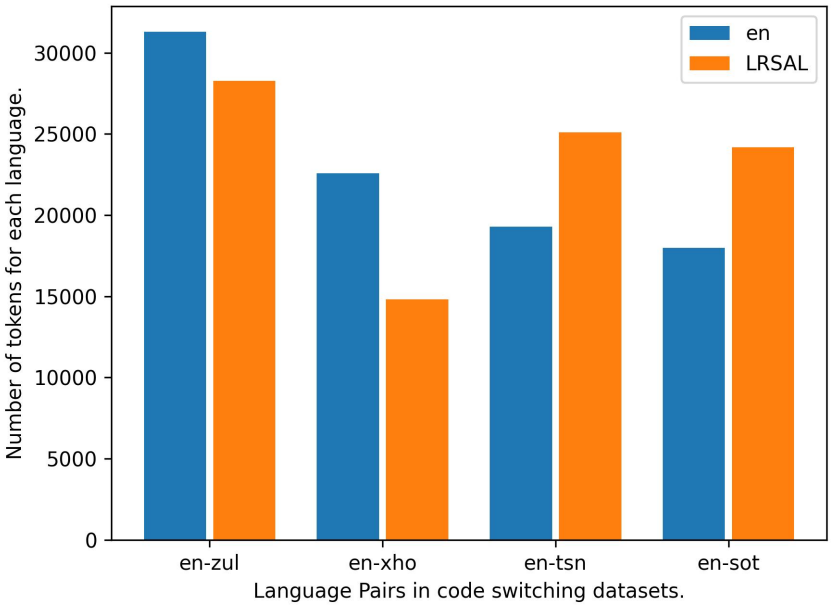
We used the labeled code-switching dataset sourced from South African soap operas [25]. This dataset covers a continuum of spontaneous code-switching types, such as intersentential, and intrasentential, and insertion. The languages covered in this dataset are as follows: English-isiZulu, English-isiXhosa, English-Sesotho, and English-Setswana.

[5] https://mlg.cput.ac.za/

[6] https://github.com/dsfsi/za-mafoko?tab=readme-ov-file

Figure 2 shows the switching distribution of each pair in the datasets. Concretely, the figure describes the number of tokens/words for each language in the code-switched dataset.

Fig. 2: Code distributions in code-switched datasets [25].



3.4 Pre-processing

A manual inspection of individual sources of monolingual corpora indicated that sources such as WMT, and MC4 contain foreign sentences and warrant further cleaning. For this, random samples were continuously drawn from the datasets for manual inspection. These samples were manually inspected by team members who are L1 (first language) speakers of the language. The L1 inspector is then tasked to highlight if additional filtering should be done. In most cases, the inspector would recommend filtering, and since the monolingual corpora are large, we assumed the issue persists in the datasets. As such, all datasets underwent language identification filtering. It is important to note that this process does not affect the quality of the dataset, only the size of the dataset. To address this, we used a publicly available language identification tool - GlotLID [19], which has been shown to have superior language identification performance in [32]. Since the

monolingual corpus is arranged by language, we used GlotLID to confirm if the sentence s_i of a known corpus C_{lang} belongs to language $lang$. If GlotLID identifies s_i as not of $lang$ of corpus C_{lang}, it is discarded. Table 1, shows the number of sentences from a collection of all monolingual corpora for each language, and the remaining number of lines before and after utilizing GlotLID on the corpora. Further pre-processing included the removal of URLs, numbers, punctuations, and then lower casing all words. For the annotated code-switching dataset, no further processing was done as this dataset is cleaned and word-annotated.

3.5 Monolingual Embeddings

Monolingual embeddings are continuous vector representations of words [22]. Various studies for generating these embeddings have been proposed, with the recent FastText technique [8] showing improvement over previous methods such as Word2Vec [7], and GloVE [28] for low-resourced settings. As such we generated monolingual embeddings using FastText with dimension $d = \{50, 100, 150, 200\}$. For English, we used the available largely pre-trained embeddings from GloVE, due to having the desired 3 dimensions $d = \{50, 100, 200\}$. To evaluate the intrinsic quality of all our embeddings, we used the SimLex-999 dataset released by [21], consisting of Setswana and Sepedi paired words together with their English translations. From this, we derived that monolingual embeddings for low-resourced languages require more data to train, as reflected by their inability to capture similar words with similar representations as compared to English representations (Appendix B.1). However, projecting the inefficiently learned representations into a shared space with an effectively learned English representation space using the 3 techniques (Canonical Correlation Analyses (CCA) [15], VecMap [5], and MUSE [20]) shows improved word similarity measures between intra-similarity (within a language) and inter-similarity (between languages) (Appendix B.2). We further calculated the Spearman's correlation of the datasets and this shows that cross-lingual embeddings outperform monolingual embeddings. An extension of this analysis is available in Appendix B.3, showing plots for all embedding dimensions and the 3 projection types to evaluate the intrinsic quality of the embeddings. We leave details in the Appendix sections as these are not the priority of this study.

Notably, lower embedding dimensions of unrelated languages (xho and zul) show higher cosine scores for paired word similarity evaluation.

3.6 Cross-lingual Embeddings

Cross-lingual embeddings are shared vector representations generated using projection techniques that aim to join monolingual embeddings of two or more languages together with the objective of transferring common desirable linguistic properties [23]. Projection techniques develop a mathematical model driven by available supervision resources, such as bilingual lexicons, parallel sentences, objective functions, etc. that aims to learn how to translate source embeddings (typically of high-resourced language) to target embeddings (of low-resourced) by transforming (shift, distort, etc.) the source embeddings into a

[7] https://code.google.com/p/word2vec/

shared vector space. A comprehensive survey detailing various challenges, opportunities, future works, and applications of cross-lingual embeddings on downstream tasks can be found in [31]. In this study, we compared three pioneering projection techniques – Canonical Correlation Analyses (CCA), VecMap, and MUSE to generate cross-lingual shared representations.

3.7 Pre-trained Models

Pre-trained multilingual models such as mBERT [12], RemBERT, XLM-r [10], and their Afro-centric counterparts: Afri-BERTa [26], Afro-XLM-r [4] have shown astonishing results for many downstream tasks such as NER, POS Tagging, Machine Translation, etc., with Afro-centric methods having a slight performance edge over massively pre-trained multilingual models with minimal to no exposure to African languages. As such, we will only concentrate on the Afro-centric model Afro-XLM-r (base and large), and the recent model Serengeti [1] due to their high performance gains on downstream tasks.

3.8 Injecting pre-trained multilingual models with explicit cross-lingual embeddings

Suppose we know the languages L_g = $\{l1, l2, ..., ln\}$ of the code-switched corpus C_{cs}. We know that the interchangeable use of linguistic tokens $t_i, ..., t_k$ from $\{l1, ..ln\}$ is not random but rather orchestrated meaningfully to bring the targeted languages coherently together. Therefore, we hypothesize that creating shared cross-lingual representations of the known target languages that meaningfully tie the target languages semantically together may improve code-switching processing. Concretely, this study's code-switched datasets are a switching condition between eng and the four languages sot, tsn, xho, and zul. Hence, for known pairs (e.g $eng - sot$), we could create shared representations between eng, and sot resulting into a meaningful shared vector space, such that translation pairs between the two languages are semantically connected (i.e translation pairs between the two languages having similar representations), which could reflect in better processing of the target code-switching dataset such as improved handling of missing vocabulary. To evaluate this theory, we devised a technique that explores the use of cross-lingual embeddings into transformer architectures by replacing the embeddings layer with the new shared representation of the known pairs. Monolingual embeddings (to model an instance where the languages are not brought closer), and cross-lingual embeddings (to model a case where languages are brought closer together) will be used to replace the embedding layer of the transformer architectures. Experiments of these two setups will be compared and contrasted.

Dimensionality Reduction and Expansion Our cross-lingual embeddings dimensions of d = $\{50, 100, 150, 200\}$ are significantly lower than the embeddings of AfroXLMr, and Serengeti of 700+. Therefore, we are tasked with either reducing the embeddings of the transformer when combined with cross-lingual embeddings or expanding cross-lingual embeddings to match existing transformer embeddings. In this case, we experimented with randomly initialized paddings with a normal distribution to expand the cross-lingual

embeddings to match transformer dimensions. For dimensionality reduction, Principal Component Analysis (PCA), Uniform Manifold Approximation and Projection, or other techniques can be adopted to reduce the existing transformer embeddings to match the static embeddings' shape. Due to limited space, we leave this for future work.

3.9 Experimental Design

Corpora sizes Following the pre-processing step, we extracted 80% of the most frequent words to generate our word embeddings.

Code-Switched dataset We explored various setups to divide the dataset into train, development, and test sets. The conventional setup included having a uniform train set across all languages of 4242 sentences and a test size of 1000 sentences, with only varying development sizes, since the size of the datasets for each language was not equal. The second setup included combining the dataset of the previous step to explore transfer learning. We combined code-switched datasets containing closely related languages (xho and zul) – two Nguni languages, and (sot and tsn) – two Sotho-Tswana languages. Finally, for each group, we added, a new language coming from a different language family, and then our last setup combined all languages.

Monolingual Embeddings generation Our monolingual embeddings were trained for 50 epochs, with mostly default set-ups for FastText except for min character and maximum character considerations of 1, and 5, respectively, attained empirically. Furthermore, all words were reduced to lowercase.

Cross-lingual Embeddings generation All projection techniques CCA, VecMap, and MUSE use a supervised projection setup with all available lexicons in this study. We used the default hyperparameter setup recommended by each technique's proposed paper since no available resources exist to evaluate the intrinsic quality of cross-lingual representations.

3.10 Pre-trained Models

Each pre-trained model was trained for 20 epochs, used a batch size of 16, a maximum sequence cut-off of 200, a learning rate of $5\exp^{-5}$ following [2], and [13]. However, Afro-XLMr-large did not perform well for the setup, and its hyperparameters were changed to a learning rate of $2\exp^{-5}$, a batch size of 32, and was trained for 10 epochs.

4 Results

This section discusses the results of this study's experimental findings. We report F1 score instead of accuracy since there is a clear class imbalance between the codes within our datasets (Figure 2). F1 score is the harmonic mean of the model's precision (Equation 1) and recall (Equation 2), defined in Equation 3 below:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1\_score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3}$$

where TP, FP, FN denote the true positive, false positive, and false negative predictions, respectively.

4.1 Baselines

Table 2, reports the F1 score performance of our baseline models: Afro-XLM-r base (b), Afro-XLM-r large (l), and Serengeti. Our results show that the models perform on par for word-based code-switching detection. Notably, increasing the dataset size by combining datasets (e.g, combining engxho with engzul to make engxhozul) shows to have a contradicting impact on code-switching detection. Firstly, combining datasets shows a drop in performance compared to monolingual training. Secondly, training a three pair of either 2 Nguni and one Sotho-Tswana or vice versa shows a 1-point drop when training with two Nguni (xho, zul), and one Sotho-Tswana (tsn), compared to training with only the two Nguni languages. We hypothesize that the addition of a language from a different language family creates an interference in the internal representation largely skewed to Nguni morphology, which negatively impacts the model's performance. Conversely, the Sotho-Tswana and single Nguni trio illustrate a performance gain in this setup. Since this pattern occurs on all models, it may imply that the internal structures of Sotho-Tswana may be robust and less susceptible to interference compared to Nguni language learning. However, we leave this investigation for future work and continue investigating transfer learning. Lastly, Afro-xlmr-base performs on par with Serengeti, while the Afro-xlmr-large model slightly lags behind with mostly 1 percentage point below across all datasets.

Table 2: Baseline model F1-scores for code-switching detection across datasets, averaged over 5 runs.

Baselines	Code-Switching dataset with base (eng)										
	xho	zul	sot	tsn	xhozul	sottsn	xhozulsot	xhozultsn	sottsnxho	sottsnzul	xhozulsottsn
Afro-xlmr-b	98.0	96.8	90.8	92.5	92.4	83.0	92.3	92.2	86.7	87.4	87.0
Afro-xlmr-l	96.8	96.9	89.9	89.8	90.0	81.8	85.3	91.5	85.3	86.0	872
Serengeti	96.8	96.9	89.9	89.8	92.4	81.8	97.2	91.4	85.3	86.0	87.2

4.2 Baselines Transfer learning

We investigate three modes of transfer learning in this section. The first setup investigates transferring models trained with multiple combinations above onto original (non-combined) datasets. The second transfer setup investigates language family grouping, where languages belonging to the same family (e.g xho and zul) are grouped into one label $ngun$, and $sttn$ for Sotho-Tswana (results reported Table 3). Surprisingly, Afro-XLMR-b performs better than Serengeti on average for family-based-grouping code-switching detection. Additionally, all 3 models show higher performance for Nguni combinations over Sotho-Tswana combination datasets. In the last setup, however, we changed the datasets to only have two labels (eng, and $swtc$), by replacing any other label that is not eng to $swtc$. This is done, to investigate if transfer could be easily modeled if the label set is reduced in a multi-code switching dataset. The results for the aforementioned last experiments are reported in Table 4. From the onset, the results show performance gains on language combinations compared to baseline results in Table 2. We hypothesize that this happens because the models may find it easy to create two representational clusters of the code-switching as opposed to creating multiple representing each language. Additionally, closely related languages may not cause incorrect predictions as in the conventional setup, as these are viewed as the same class $swtc$. This is also supported by higher scores for the original dataset scores. Regardless, this approach may be a better alternative to code-switching detection as many African languages are not effectively supported at the onset (i.e. when pre-training these large multilingual models), where, instead we model code-switching as certainty (eng) and uncertainty (not eng) binary classification with the assumption that English will easily be detected with high confidence (i.e. due to high prevalence of large datasets used for pre-training). With this setup, smaller models such as Naive Bayes, SVM, etc, can be used for language identification of the $swtc$ tag for finer-grained detail extraction, thus breaking the task into two subsequent tasks. However, this claim requires additional empirical evidence and we leave it for future works. Finally, on average, Serengeti performs better compared to the two variations of Afro-XLM-r for this transfer category.

Table 3: Reports model's F1-score for code-switching detection for each dataset averaged over 5 runs of grouped label datasets. The xho, zul are grouped into Nguni, and sot tsn grouped into Sotho-Tswana (sttsn).

Transfer	Code-Switching dataset with base (eng) and labels ngun and sttsn						
	xhozul	sottsn	xhozulsot	xhozultsn	sottsnxho	sottsnzul	xhozulsottsn
Afro-xlmr-b	97.8	92.4	94.9	95.7	93.8	93.8	94.6
Afro-xlmr-l	97.4	91.9	94.9	95.6	93.6	93.4	94.3
Serengeti	97.4	91.9	94.9	95.7	93.6	93.4	94.3

Table 4: Reports the label change transfer model's F1-score for code-switching detection for each dataset averaged over 5 runs.

Transfer	Code-Switching dataset with base (eng) and second label switched (swtc)						
	xhozul	sottsn	xhozulsot	xhozultsn	sottsnxho	sottsnzul	xhozulsottsn
Afro-xlmr-b	85.9	93.1	95.4	96.0	93.6	94.0	95.1
Afro-xlmr-l	97.9	92.5	95.5	96.1	94.1	80.7	95.1
Serengeti	97.6	92.2	95.1	95.9	93.8	93.7	94.8

4.3 Injecting pre-trained multilingual models with explicit cross-lingual embeddings

Table 5, shows the experimental results of this study using various cross-lingual embeddings settings. Due to limited space, we only reported the results of Afro-XLM-r-base. We presented the remaining results in Appendix C with accompanying discussions and the Serengeti model results. From these results, CCA and VecMap embeddings show better performance for lower dimensions 50 and 100 over Muse embeddings, while Muse surpasses these two techniques on the highest dimension 200 on the original datasets xho, zul, sot, and tsn. This behavior is not clear as to why it occurs, as we expected Muse embeddings to perform consistently better than the other two techniques, as shown by its representation quality illustrated in the intrinsic evaluation (Appendix B.2, Appendix B.3). This could mean that intrinsic performance is not correlated to extrinsic performance for this task. On the combination datasets, only Muse embeddings show consistent performance, while the performance of CCA and VecMap fluctuates depending on the data combinations. This could be due to that, while the English to low-resourced language ($en - LRL_i$) pair remains semantically connected through the projection, the inclusion of a foreign pair $en - LRL_k$ in the same model introduces the same issue (i.e. the disconnect) we are trying to solve in the embedding space (i.e. the attempt to bring representation of the code-switched languages closer together), for i, and k being low-resource languages. For example, combining en-tsn, with en-xho datasets, results in interference between tsn and xho as these were not explicitly joined through cross-lingual projection techniques. Therefore, there is no common space for the two pairs, thus creating the very issue we are trying to solve.

To understand the impact of these injected cross-lingual embeddings we trained the best-performing model - Afro-XLMr-base, with monolingual embeddings from which these joined representations were formed. Table 16, shows these results for non-projected embeddings of xho, zul, sot, tsn created using FastText. From this, we can see that monolingual performance is significantly lower for all of the various evaluation metrics: Accuracy, Precision, Recall, and F1-score. This, in a way, justifies our hypothesis that, perhaps not treating the codes as independent spaces but rather semantically joining their individual spaces at pre-training could improve code-switching processing as these languages are brought to the same shared space.

From these results collectively, it is plausible that the high performance of baseline pre-trained models can be due to the implicit multi-lingual representation learned over vast amounts of textual data compared to the small samples used to train our cross-lingual

Table 5: Reports the Afro-XLMr-base F1-score for code-switching detection for each dataset, averaged over 5 runs using the three embedding techniques. The injected embeddings were randomly selected from the full embedding matrix to match the vocabulary size of the transformer.

Models	Code-Switching dataset with base (eng)										
	xho	zul	sot	tsn	xhozul	sottsn	xhozulsot	xhozultsn	sottsnxho	sottsnzul	xhozulsottsn
Embedding Dimension: 50											
CCA	81.5	84.6	77.1	79.5	57.1	56.7	73.8	61.5	66.5	77.6	78.7
VecMap	75.7	84.0	85.3	83.9	80.7	64.7	83.2	76.4	67.1	63.1	45.0
Muse	76.2	79.6	62.8	68.1	70.5	70.4	73.1	74.3	59.3	62.1	69.7
Embedding Dimension: 100											
CCA	81.3	83.2	76.0	66.5	69.1	74.8	74.1	75.8	79.6	78.2	56.5
VecMap	81.5	84.1	78.1	84.7	69.4	74.3	82.5	45.7	75.5	68.0	62.0
Muse	67.0	81.0	76.1	74.1	72.1	70.9	74.2	64.4	70.4	71.9	70.0
Embedding Dimension: 200											
CCA	71.3	61.2	77.6	71.8	68.1	66.8	48.5	71.3	79.7	72.1	59.7
VecMap	71.2	84.0	79.5	75.7	72.5	75.2	79.3	69.6	78.1	59.4	77.9
Muse	76.1	79.6	76.6	74.9	71.4	71.1	74.4	75.5	71.9	71.7	71.7

Table 6: Reports the Afro-XLMr-base Accuracy (acc), Precision (prec), Recall (rec), and F1-score (f1) for code-switching detection for each dataset averaged over 5 runs using FastText Monolingual embeddings and Glove embeddings for English words. The injected embeddings were randomly selected from the full embedding matrix to match the vocabulary size of the transformer.

Metric	Code-Switching dataset with base (eng)										
	xho	zul	sot	tsn	xhozul	sottsn	xhozulsot	xhozultsn	sottsnxho	sottsnzul	xhozulsottsn
Embedding Dimension: 50											
acc	93.4	81.0	90.8	90.8	80.5	82.4	86.8	82.8	66.7	76.2	78.6
prec	71.4	56.3	60.5	68.8	55.6	63.5	63.4	60.6	47.4	57.1	59.3
rec	68.1	52.5	61.7	64.9	56.2	62.4	64.2	61.3	40.5	54.3	61.3
f1	69.1	53.6	61.1	66.6	55.7	62.3	63.6	60.7	42.7	55.1	59.9
Embedding Dimension: 100											
acc	92.9	86.7	71.8	78.6	81.5	75.5	81.3	74.0	78.4	76.2	75.0
prec	63.2	57.7	57.4	60.5	58.3	54.8	55.4	52.4	58.9	58.8	54.6
rec	63.6	55.9	44.3	54.1	59.7	55.4	49.4	51.2	61.3	54.7	51.9
f1	63.4	56.3	49.1	56.9	58.7	54.6	51.6	51.3	59.7	55.8	52.4
Embedding Dimension: 200											
acc	92.4	80.9	75.7	90.6	72.2	82.4	65.3	65.0	55.3	81.5	75.3
prec	66.4	56.2	56.0	71.8	46.1	62.6	45.1	43.7	36.5	61.5	56.3
rec	64.1	52.4	32.5	66.2	44.1	61.6	40.6	39.7	27.7	62.8	53.5
f1	64.6	53.5	41.1	68.7	44.7	61.4	42.1	40.9	30.7	61.7	54.1

embeddings. Regardless, a clear trend emerges, that from monolingual embeddings, a boost in performance can be attained by using cross-lingual embeddings, and further improvements is attained by using largely trained multilingual embeddings.

5 Analyses and Discussion

We also aimed to investigate the extent to which shared representations support text processing, especially for unknown or out-of-vocabulary words for improved performance. That is, in cases where vocabulary is missing, we want to analyze if other related words are used for improving performance and if this is traced to shared representations. We aimed to achieve this through embedding attention score analyses. This is done by plotting embedding attention scores of predicted sentences to see where priority is placed for certain words.

Sentence attention score analyses consider relationships of words within a sentence, we also wanted to consider alternative words within the entire embedding matrix that may be important in processing the final output of the layer. We observed that high attention was also given to words outside the main sentence vocabulary. This could mean that, related words, were identified (possibly made possible by shared representations), thus, improving the processing of the target text. Monolingual plots do not show this behavior of exterior attention. This implies that, indeed, deeper connections may have been forged through semantically connecting monolingual embeddings. Further analyses on this is provided in Appendix D.

6 Conclusion

Code-switching has gained research attention in the field of Natural Language Processing, especially for low-resourced languages due to most communities being largely multilingual. Large pre-trained multilingual models are typically a de facto standard processing tool for many downstream tasks due to their increased processing capabilities. However, the use of explicit cross-lingual embeddings to bridge the gap between the language representations of the known codes lags behind. In this study, we explore the use of cross-lingual embeddings that aim to bring known language pairs closer together to efficiently learn code-switching detection. Indeed, our experimental analyses show that mapping the switched languages into a single shared vector space before training shows untapped processing capabilities for code-switching detection. Concretely, fine-tuning AfroXLM-r, and Serengeti architectures with explicit cross-lingual embeddings outperforms monolingual (only joined through concatenation) embeddings. Although the cross-lingual injection experiments performed poorly compared to the original multilingual embedding baselines, our results simply imply that, learning explicit shared representation between known codes, formed a high-order representational space, enhancing inter-learning between token subspaces, allowing efficient processing of code-switched text.

Limitations

The vocabulary of the embeddings is larger than the transformer embeddings. This means the selection of the appropriate words to add to the model embedding becomes crucial.

We explored a theoretically ideal scenario for generating cross-lingual embeddings and have not explored set-ups such as how many bilingual lexicons' signals are sufficient for generating the best shared representations.

In line with the above limitation, this study did not explore hyperparameter fine-tuning for injected cross-lingual embeddings. We advise future works to consider this as it may significantly improve outcomes.

Our experiments did not consider the many massively pre-trained multilingual models, such as RemBert, mBERT, including afro-centric pre-trained models such as AfroLM, AfriBerta, etc. This is due to limitations is computational power as well as time constraints as recreating and injecting explicit cross-lingual embeddings takes more time, even in the advent of multiprocessing capabilities to speed up the process.

Ethics Statement

Data and Models Disclaimer The datasets and models used in this study are collected from publicly available resources with no potential harm, threats, and risks to society.

License

Acknowledgements

The authors gratefully acknowledge the support of the ABSA Chair of Data Science and the Data Science for Social Impact (DSFSI) Lab at the University of Pretoria. This work was supported by UK International Development and the International Development Research Centre (IDRC), Ottawa, Canada, under the AI4D Africa Program. DSFSI also acknowledges gifts from NVIDIA, Google.org, OpenAI and Meta.

References

1. Adebara, I., Elmadany, A., Abdul-Mageed, M., Alcoba Inciarte, A.: SERENGETI: Massively multilingual language models for Africa. In: Findings of the Association for Computational Linguistics: ACL 2023. pp. 1498–1537. Association for Computational Linguistics, Toronto, Canada (Jul 2023). https://doi.org/10.18653/v1/2023.findings-acl.97, https://aclanthology.org/2023.findings-acl.97

2. Adelani, D.I., Abbott, J., Neubig, G., D'souza, D., Kreutzer, J., Lignos, C., Palen-Michel, C., Buzaaba, H., Rijhwani, S., Ruder, S., et al.: Masakhaner: Named entity recognition for african languages. Transactions of the Association for Computational Linguistics **9**, 1116–1131 (2021)
3. Aguilar, G., AlGhamdi, F., Soto, V., Diab, M., Hirschberg, J., Solorio, T.: Named entity recognition on code-switched data: Overview of the calcs 2018 shared task. arXiv preprint arXiv:1906.04138 (2019)
4. Alabi, J.O., Adelani, D.I., Mosbach, M., Klakow, D.: Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In: Proceedings of the 29th International Conference on Computational Linguistics. pp. 4336–4349. International Committee on Computational Linguistics, Gyeongju, Republic of Korea (Oct 2022), https://aclanthology.org/2022.coling-1.382
5. Artetxe, M., Labaka, G., Agirre, E.: A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. arXiv preprint arXiv:1805.06297 (2018)
6. Attia, M., Samih, Y., Maier, W.: Ghht at calcs 2018: Named entity recognition for dialectal arabic using neural networks. In: Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching. pp. 98–102 (2018)
7. Barman, U., Wagner, J., Foster, J.: Part-of-speech tagging of code-mixed social media content: Pipeline, stacking and joint modelling. In: Proceedings of the second workshop on computational approaches to code switching. pp. 30–39 (2016)
8. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the association for computational linguistics **5**, 135–146 (2017)
9. Chittaranjan, G., Vyas, Y., Bali, K., Choudhury, M.: Word-level language identification using crf: Code-switching shared task report of msr india system. In: Proceedings of the first workshop on computational approaches to code switching. pp. 73–79 (2014)
10. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116 (2019)
11. Costa-jussà, M.R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., et al.: No language left behind: Scaling human-centered machine translation. arXiv preprint arXiv:2207.04672 (2022)
12. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
13. Dione, C.M.B., Adelani, D., Nabende, P., Alabi, J., Sindane, T., Buzaaba, H., Muhammad, S.H., Emezue, C.C., Ogayo, P., Aremu, A., et al.: Masakhapos: Part-of-speech tagging for typologically diverse african languages. arXiv preprint arXiv:2305.13989 (2023)
14. Eiselen, R., Puttkammer, M.J.: Developing text resources for ten south african languages. In: LREC. pp. 3698–3703. Citeseer (2014)
15. Faruqui, M., Dyer, C.: Improving vector space word representations using multilingual correlation. In: Proceedings of EACL (2014)
16. Van der Goot, R., Çetinoğlu, Ö.: Lexical normalization for code-switched data and its effect on pos-tagging. arXiv preprint arXiv:2006.01175 (2020)
17. Hussain, A., Arshad, M.U.: An attention based neural network for code switching detection: English & roman urdu. arXiv preprint arXiv:2103.02252 (2021)
18. Jose, N., Chakravarthi, B.R., Suryawanshi, S., Sherly, E., McCrae, J.P.: A survey of current datasets for code-switching research. In: 2020 6th international conference on advanced computing and communication systems (ICACCS). pp. 136–141. IEEE (2020)
19. Kargaran, A.H., Imani, A., Yvon, F., Schütze, H.: Glotlid: Language identification for low-resource languages. In: The 2023 Conference on Empirical Methods in Natural Language Processing (2023), https://openreview.net/forum?id=dl4e3EBz5j

20. Lample, G., Conneau, A., Denoyer, L., Ranzato, M.: Unsupervised machine translation using monolingual corpora only. arXiv preprint arXiv:1711.00043 (2017)
21. Makgatho, M., Marivate, V., Sefara, T., Wagner, V.: Training cross-lingual embeddings for setswana and sepedi. arXiv preprint arXiv:2111.06230 (2021)
22. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
23. Mikolov, T., Le, Q.V., Sutskever, I.: Exploiting similarities among languages for machine translation. arXiv preprint arXiv:1309.4168 (2013)
24. Modipa, T.I., De Wet, F., Davel, M.H.: Implications of sepedi/english code switching for asr systems (2013)
25. Niesler, T., et al.: A first south african corpus of multilingual code-switched soap opera speech. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (2018)
26. Ogueji, K., Zhu, Y., Lin, J.: Small data? no problem! exploring the viability of pretrained multilingual language models for low-resourced languages. In: Proceedings of the 1st Workshop on Multilingual Representation Learning. pp. 116–126. Association for Computational Linguistics, Punta Cana, Dominican Republic (Nov 2021), https://aclanthology.org/2021.mrl-1.11
27. Patro, J., Samanta, B., Singh, S., Basu, A., Mukherjee, P., Choudhury, M., Mukherjee, A.: All that is english may be hindi: Enhancing language identification through automatic ranking of likeliness of word borrowing in social media. arXiv preprint arXiv:1707.08446 (2017)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
29. of Pretoria, U.: Open educational resource term bank, https://www.up.ac.za/african-languages/news/post_2728581-open-educational-resource-term-bank-pg2
30. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv e-prints (2019)
31. Sindane, T., Marivate, V., Modupe, A.: Cross-lingual embedding methods and applications: A systematic review for low-resourced scenarios. Natural Language Processing Journal **12**, 100157 (2025). https://doi.org/https://doi.org/10.1016/j.nlp.2025.100157, https://www.sciencedirect.com/science/article/pii/S2949719125000330
32. Sindane, T.A., Marivate, V.: From n-grams to pre-trained multilingual models for language identification. In: Proceedings of the 4th International Conference on Natural Language Processing for Digital Humanities. pp. 229–239 (2024)
33. Singh, K., Sen, I., Kumaraguru, P.: A twitter corpus for hindi-english code mixed pos tagging. In: Proceedings of the sixth international workshop on natural language processing for social media. pp. 12–17 (2018)
34. Sitaram, S., Chandu, K.R., Rallabandi, S.K., Black, A.W.: A survey of code-switched speech and language processing. arXiv preprint arXiv:1904.00784 (2019)
35. Solorio, T., Liu, Y.: Part-of-speech tagging for english-spanish code-switched text. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. pp. 1051–1060 (2008)
36. Vegi, P., J, S., Paul, B., Mishra, A., Banjare, P., K R, P.K., Viswanathan, C.: Webcrawl african : A multilingual parallel corpora for african languages. In: Proceedings of the Seventh Conference on Machine Translation. pp. 1076–1089. Association for Computational Linguistics, Abu Dhabi (December 2022), https://aclanthology.org/2022.wmt-1.105
37. Vyas, Y., Gella, S., Sharma, J., Bali, K., Choudhury, M.: Pos tagging of english-hindi code-mixed social media content. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 974–979 (2014)

38. Winata, G.I., Lin, Z., Fung, P.: Learning multilingual meta-embeddings for code-switching named entity recognition. In: Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019). pp. 181–186 (2019)
39. Winata, G.I., Wu, C.S., Madotto, A., Fung, P.: Bilingual character representation for efficiently addressing out-of-vocabulary words in code-switching named entity recognition. arXiv preprint arXiv:1805.12061 (2018)
40. Xia, M.X.: Codeswitching language identification using subword information enriched word vectors. In: Proceedings of the second workshop on computational approaches to code switching. pp. 132–136 (2016)

A Monolingual embeddings visualization

Visualizing the quality of monolingual embeddings can be a challenging task. In this Appendix, we present three visualization techniques: Principal Component Analyses (PCA), Uniform Manifold Approximation and Projection (UMAP), and t-distributed Stochastic Neighbor Embedding (t-SNE) for visualizing monolingual and cross-lingual embeddings in 2 dimensions. We present for each technique, visual plots for embeddings of 50, 100, 150, and 200 dimensions. For each pair of languages we concatenated the embeddings of the two languages and for words that appear in both vector spaces of the two languages we used the average of the two vectors as the vector representation of the duplicate word. Visualising all embeddings of word vectors in one plots causes difficulties in interpreting the word vectors relations and semantics, and such we created five clusters from the embeddings and only plotted a sample of 10 words from the clusters to get a sense of what the clusters contain. Figs. 3 to 8, Figs. 10 to 15, and Figs. 17 to 22 show the PCA, UMAP, and t-SNE plots when using monolingual embeddings respectively. Figs. 24 to 29, Figs. 31 to 36, and Figs. 38 to 43 show the PCA, UMAP, and t-SNE plots when using canonical correlation analysis (CCA) projection embeddings respectively, and Figs. 45 to 50, Figs. 52 to 57, and Figs. 59 to 64 show the PCA, UMAP, and t-SNE plots when using UMAP projection embeddings respectively. While Figs. 66 to 71, Figs. 73 to 78, and Figs. 80 to 85 show the PCA, UMAP, and t-SNE plots when using monolingual embeddings respectively. All three techniques were able to create clearly separable clusters with PCA showing more readable words within the clusters unlike UMAP and t-SNE. More importanlty, word relations are captured within the embeddings clusters. For example, similar words such as 'apere' –translation 'wore', and 'apara' – translation 'wear'; 'babedi' –translation 'the two' or 'couple', and 'bedi' translation 'twice', are captured in the same cluster.

B Cosine Similarities of Word Vectors

Measuring the similarity of word vectors for similar and dissimilar evaluation provides insights on the intrinsic quality of the embeddings. In this Appendix, we used the available SimLex-999 dataset released by [21] for measuring how closely related word vector are represented in word embedding spaces. The dataset has word pairs for Sepedi and Setswana only. The datasets is derived from the original English SimLex-999 in which the original English words (word1 and word2) are translated to translate1 and translate2 for the two languages, Sesotho and Setswana. That is, each dataset will contain four pairs - 2 original English pairs and the 2 translated pairs together with a score for each pair to measure the similarity of the words. For our experimental purposes, we investigated the similarity of the combinations – low-resourced to low-resourced for the translated words (lr-t), high-resourced to high-resourced (hr), high-resourced to low-resourced translate 2 (hrlr-t), and low-resourced translate 1 to high-resourced word 2 (lr-rhr). Our monolingual embeddings of the four languages Setswana, Sesotho, IsiXhosa and isiZulu, were evaluated on the two available datasets of Sesotho, and Setswana. The following Section B.1, and B.2 reports the Cosine plots for the aforementioned datasets.

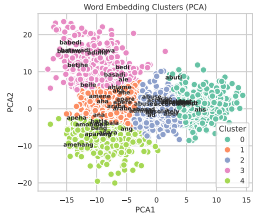


Fig. 3: en-sot Emb

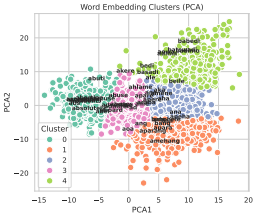


Fig. 4: en-sot Emb

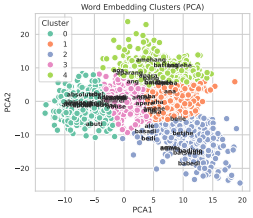


Fig. 5: en-sot Emb

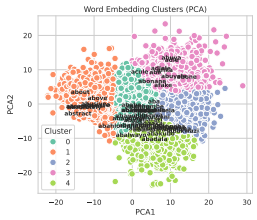


Fig. 6: en-zul Emb

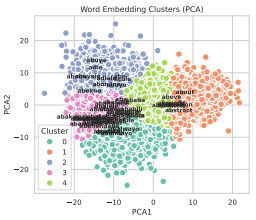


Fig. 7: en-zul Emb

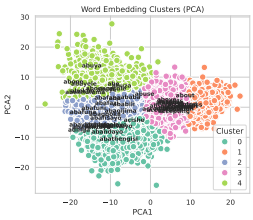


Fig. 8: en-zul Emb

Fig. 9: PCA Mono Emb plots for dimension 50 (left), 100(middle), and 200 (right)

B.1 Monolingual cosine similarity scores

Figs. 87 to 99, Figs. 101 to 113,Figs. 115 to 127, and Figs. 129 to 141 show the monolingual cosine similarity plots for four languages tsn, sot, xho, and zul, including their different dimension (50, 100, and 200). For each dataset lr-t, hr, hrlr-t, and lr-thr, we sampled 10 paired words using random sample 10, and generated the cosine plots for the 10 pairs. These plots rigorously show across multiple setups that monolingual embeddings where not able o capture word similarities sufficiently, especially when considering similar words from different languages. Cross-lingual embeddings, improves on this limitations and this is illustrated in the next subsection.

B.2 Cross-lingual cosine similarity scores

Similar to Section B.1, for each dataset lr-t, hr, hrlr-t, and lr-thr, 10 samples were extracted and the cosine similarity scores for the vector representations of the pairs were calculated. To analyse the intrinsic quality of monolingual embeddings, Figs. 87 to 141 shows the cosine similarity of paired words calculated from their vectors representations extracted from the monolingual embeddings. These figures not only show that the monolingual embeddings for the considered low-resourced languages are not able to capture similar words within a single language (i.e monolingual similarity) effectively but also fail to capture similarities across languages (cross-lingual similarity). This is could be the impact of insufficient training data for low-resourced languages, which is

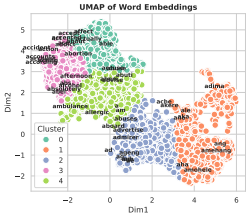


Fig. 10: en-sot Emb

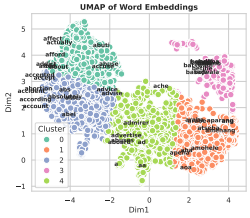


Fig. 11: en-sot Emb

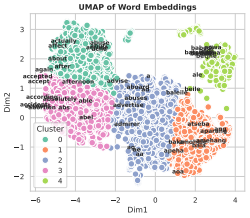


Fig. 12: en-sot Emb

Row 4

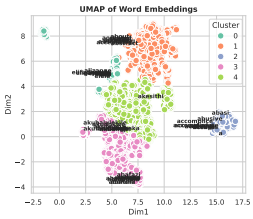


Fig. 13: en-zul Emb

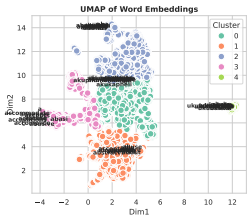


Fig. 14: en-zul Emb

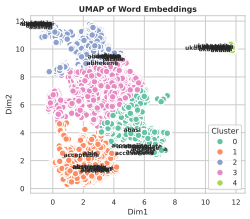


Fig. 15: en-zul Emb

Fig. 16: UMAP Mono Emb plots for dimension 50 (left), 100(middle), and 200 (right)

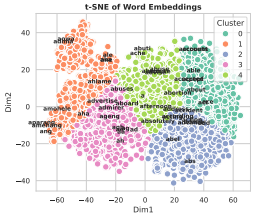


Fig. 17: en-sot Emb

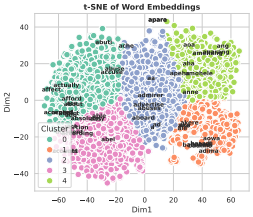


Fig. 18: en-sot Emb

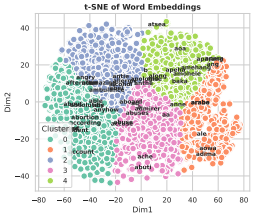


Fig. 19: en-sot Emb

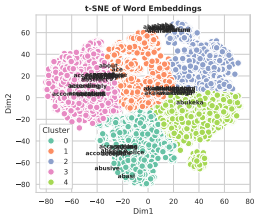


Fig. 20: en-zul Emb

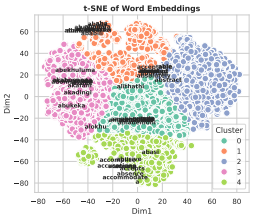


Fig. 21: en-zul Emb

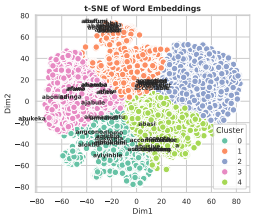


Fig. 22: en-zul Emb

Fig. 23: t-SNE Mono Emb plots for dimension 50 (left), 100(middle), and 200 (right)

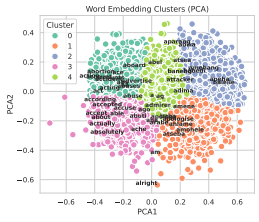


Fig. 24: en-sot Emb

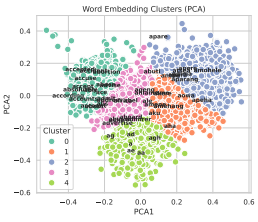


Fig. 25: en-sot Emb

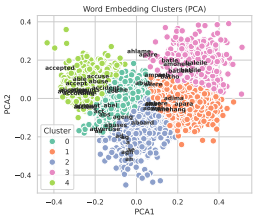


Fig. 26: en-sot Emb

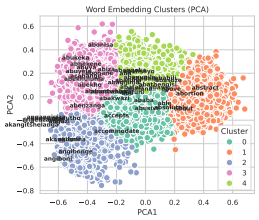


Fig. 27: en-zul Emb

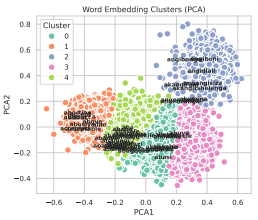


Fig. 28: en-zul Emb

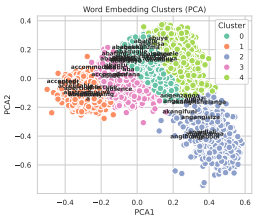


Fig. 29: en-zul Emb

Fig. 30: PCA CCA Emb plots for dimension 50 (left), 100(middle), and 200 (right)

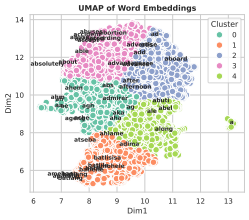


Fig. 31: en-sot Emb

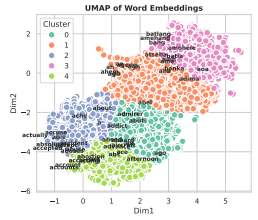


Fig. 32: en-sot Emb

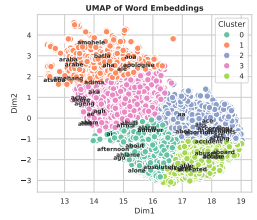


Fig. 33: en-sot Emb

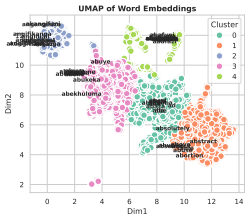


Fig. 34: en-zul Emb

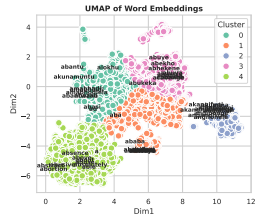


Fig. 35: en-zul Emb

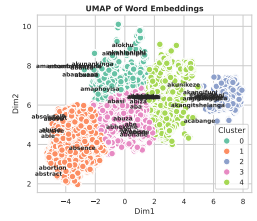


Fig. 36: en-zul Emb

Fig. 37: UMAP CCA Emb plots for dimension 50 (left), 100(middle), and 200 (right)

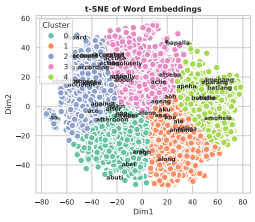


Fig. 38: en-sot Emb

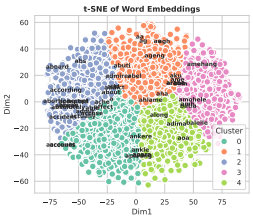


Fig. 39: en-sot Emb

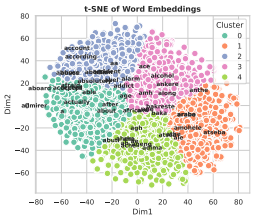


Fig. 40: en-sot Emb

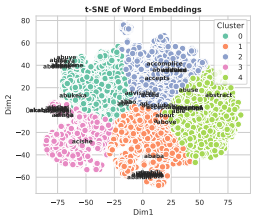


Fig. 41: en-zul Emb

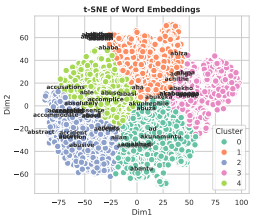


Fig. 42: en-zul Emb

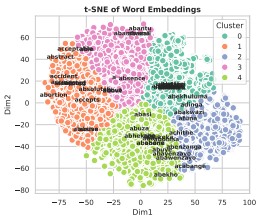


Fig. 43: en-zul Emb

Fig. 44: t-SNE CCA Emb plots for dimension 50 (left), 100(middle), and 200 (right)

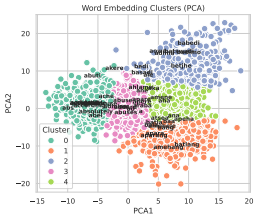


Fig. 45: en-sot Emb

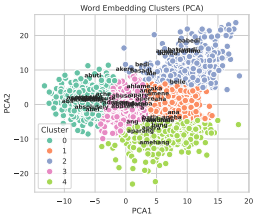


Fig. 46: en-sot Emb

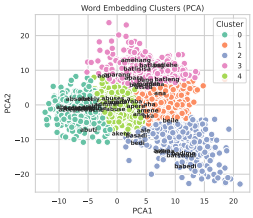


Fig. 47: en-sot Emb

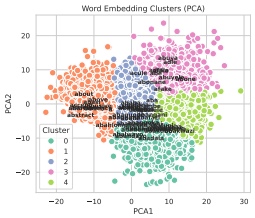


Fig. 48: en-zul Emb

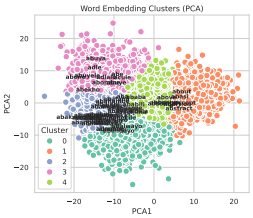


Fig. 49: en-zul Emb

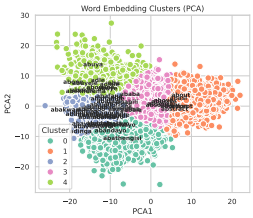


Fig. 50: en-zul Emb

Fig. 51: PCA Muse Emb plots for dimension 50 (left), 100(middle), and 200 (right)

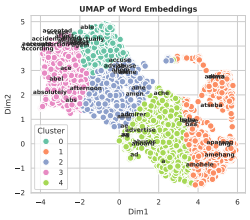


Fig. 52: en-sot Emb

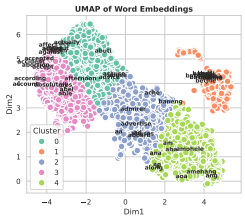


Fig. 53: en-sot Emb

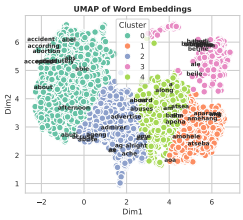


Fig. 54: en-sot Emb

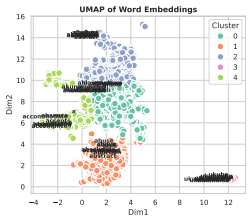


Fig. 55: en-zul Emb

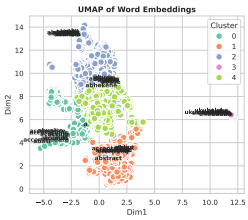


Fig. 56: en-zul Emb

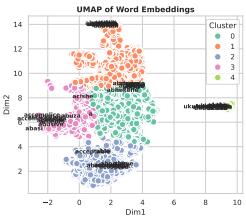


Fig. 57: en-zul Emb

Fig. 58: UMAP Muse Emb plots for dimension 50 (left), 100(middle), and 200 (right)

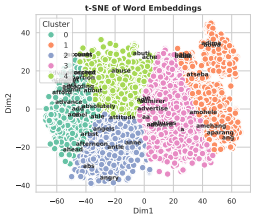


Fig. 59: en-sot Emb

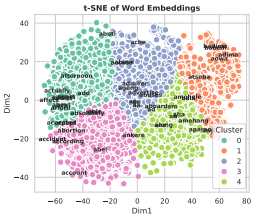


Fig. 60: en-sot Emb

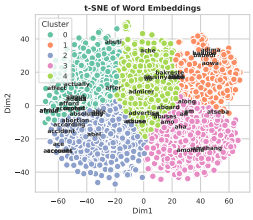


Fig. 61: en-sot Emb

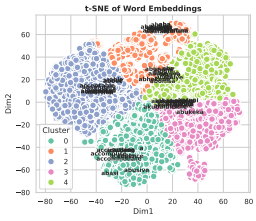


Fig. 62: en-zul Emb

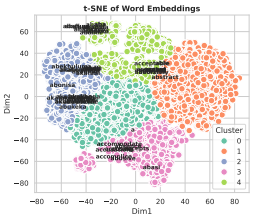


Fig. 63: en-zul Emb

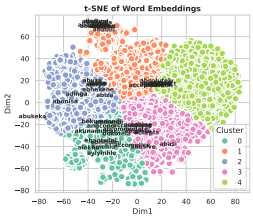


Fig. 64: en-zul Emb

Fig. 65: t-SNE Muse Emb plots for dimension 50 (left), 100(middle), and 200 (right)

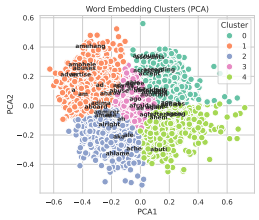


Fig. 66: en-sot Emb

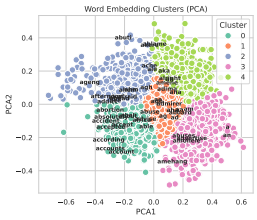


Fig. 67: en-sot Emb

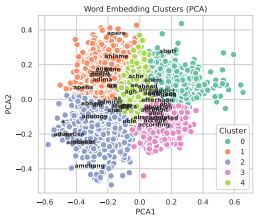


Fig. 68: en-sot Emb

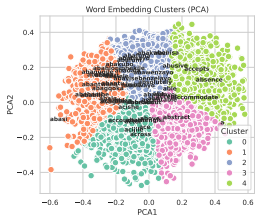


Fig. 69: en-zul Emb

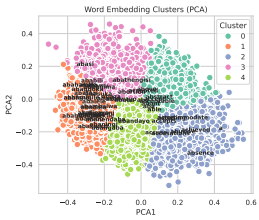


Fig. 70: en-zul Emb

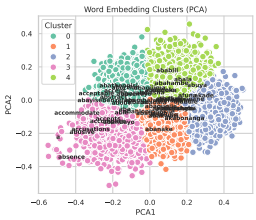


Fig. 71: en-zul Emb

Fig. 72: PCA VecMap Emb plots for dimension 50 (left), 100(middle), and 200 (right)

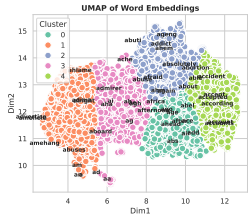


Fig. 73: en-sot Emb

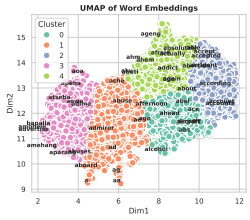


Fig. 74: en-sot Emb

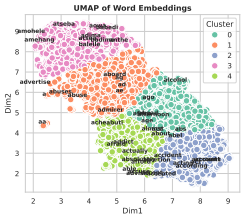


Fig. 75: en-sot Emb

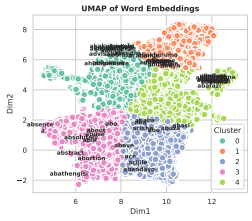


Fig. 76: en-zul Emb

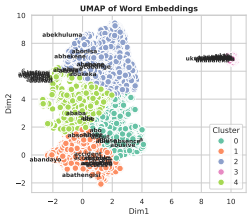


Fig. 77: en-zul Emb

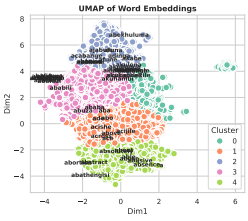


Fig. 78: en-zul Emb

Fig. 79: UMAP VecMap Emb plots for dimension 50 (left), 100(middle), and 200 (right)

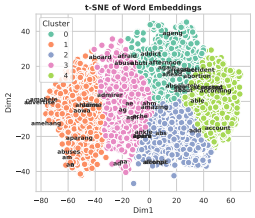


Fig. 80: en-sot Emb

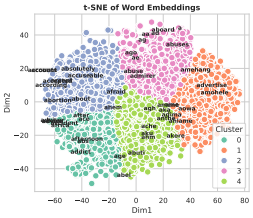


Fig. 81: en-sot Emb

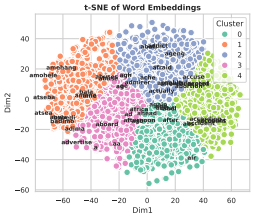


Fig. 82: en-sot Emb

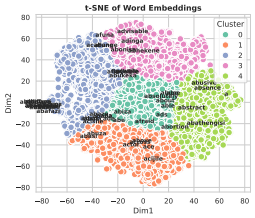


Fig. 83: en-zul Emb

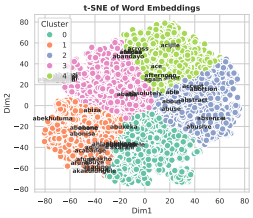


Fig. 84: en-zul Emb

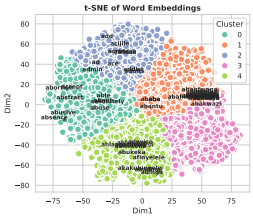


Fig. 85: en-zul Emb

Fig. 86: t-SNE VecMap Emb plots for dimension 50 (left), 100(middle), and 200 (right)

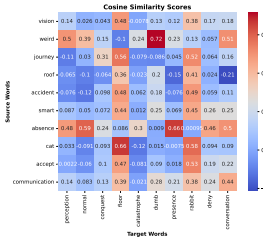


Fig. 87: hr

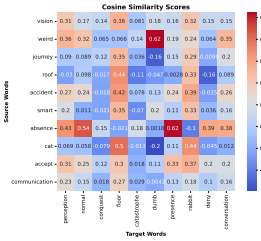


Fig. 88: hr

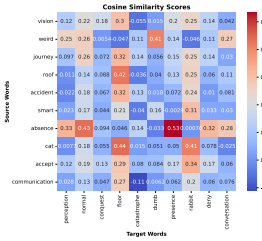


Fig. 89: hr

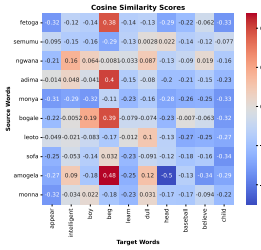


Fig. 90: lr-t and hr

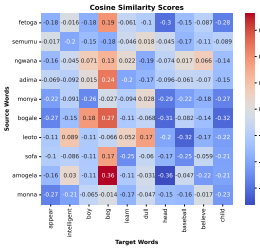


Fig. 91: lr-t and hr

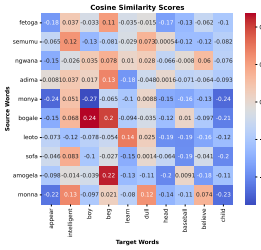


Fig. 92: lr-t and hr

Fig. 93: sot Mono Emb plots of 50 (left), 100(middle), and 200 (right), nso test

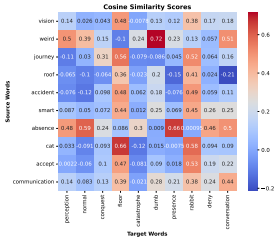


Fig. 94: hr

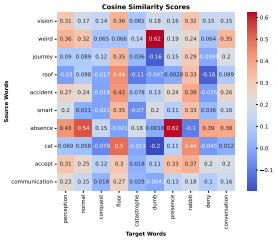


Fig. 95: hr

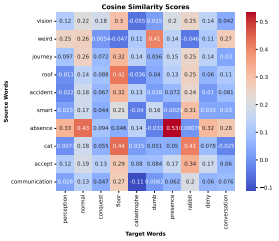


Fig. 96: hr

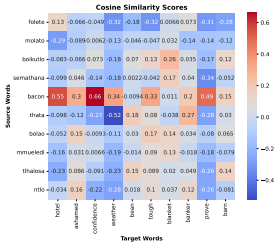


Fig. 97: lr-r and hr

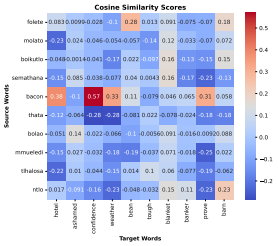


Fig. 98: lr-t and hr

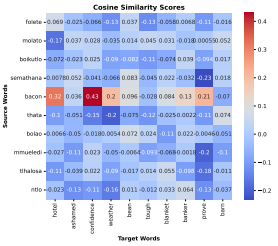


Fig. 99: lr-t and hr

Fig. 100: tsn Mono Emb plots of 50 (left), 100(middle), and 200 (right), tsn test

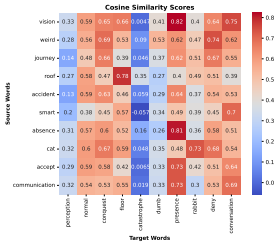


Fig. 101: hr

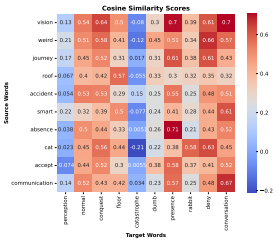


Fig. 102: hr

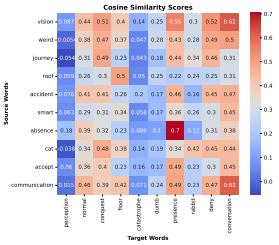


Fig. 103: hr

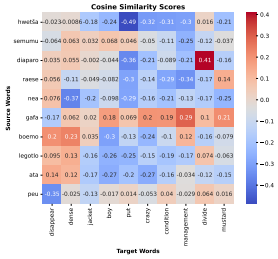


Fig. 104: lr-t and hr

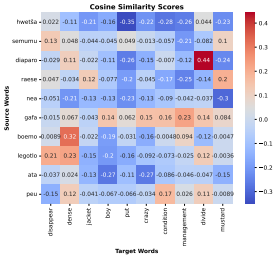


Fig. 105: lr-t and hr

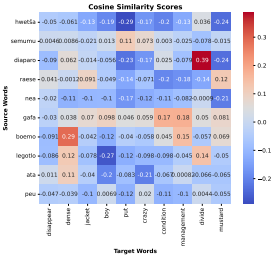


Fig. 106: lr-t and hr

Fig. 107: sot Mono Emb plots of 50 (left), 100(middle), and 200 (right), nso test

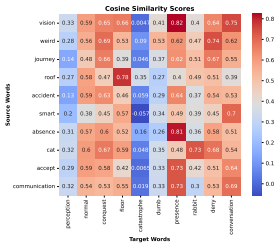


Fig. 108: hr

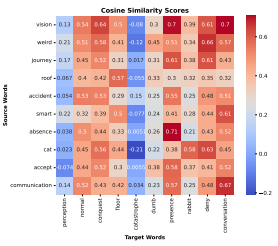


Fig. 109: hr

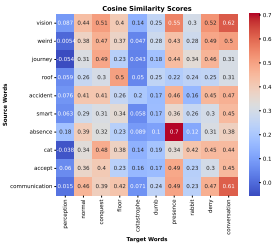


Fig. 110: hr

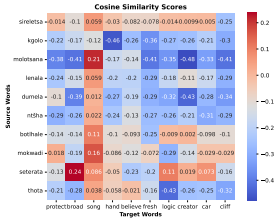


Fig. 111: lr-r and hr

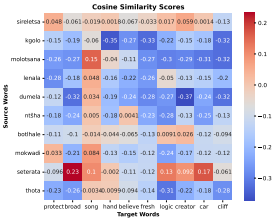


Fig. 112: lr-t and hr

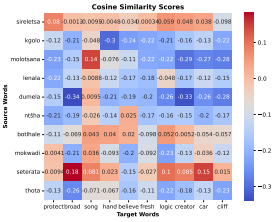


Fig. 113: lr-t and hr

Fig. 114: sot Mono Emb plots of 50 (left), 100(middle), and 200 (right), tsn test

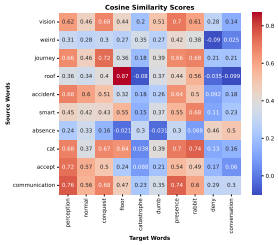


Fig. 115: hr

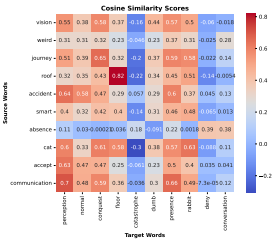


Fig. 116: hr

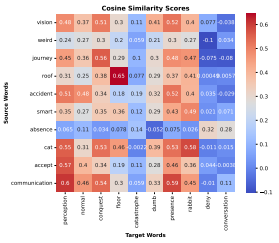


Fig. 117: hr

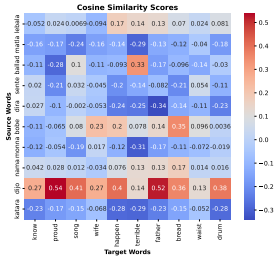


Fig. 118: lr-t and hr

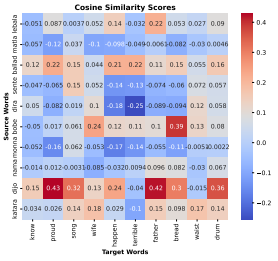


Fig. 119: lr-t and hr

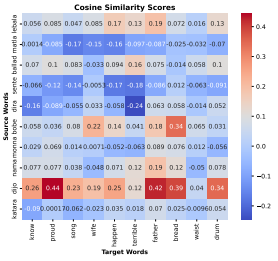


Fig. 120: lr-t and hr

Fig. 121: xho Mono Emb plots of 50 (left), 100(middle), and 200 (right), nso test

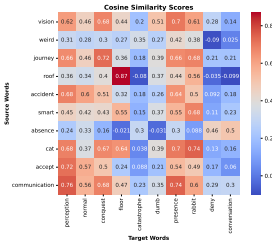


Fig. 122: hr

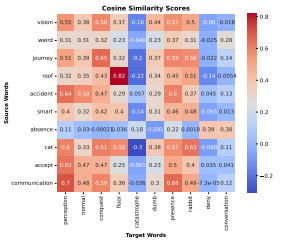


Fig. 123: hr

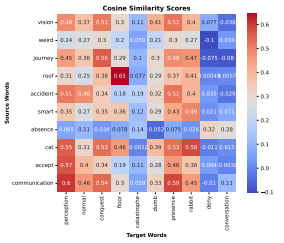


Fig. 124: hr

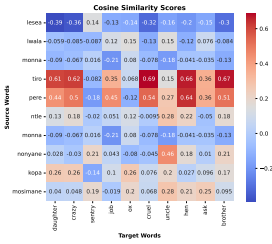


Fig. 125: lr-t and hr

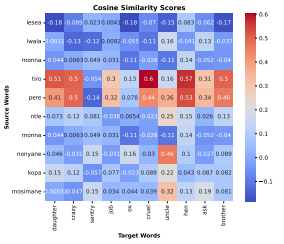


Fig. 126: lr-t and hr

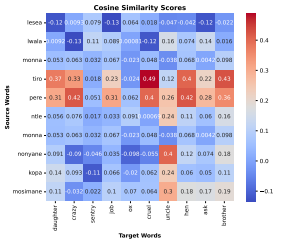


Fig. 127: lr-t and hr

Fig. 128: xho Mono Emb plots of 50 (left), 100(middle), and 200 (right), tsn test

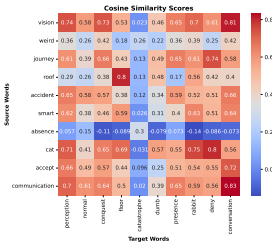


Fig. 129: hr

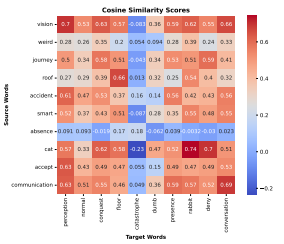


Fig. 130: hr

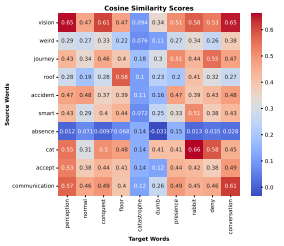


Fig. 131: hr

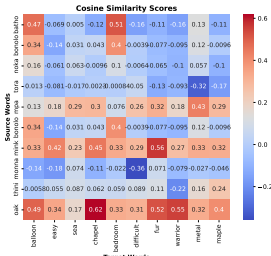


Fig. 132: lr-t and hr

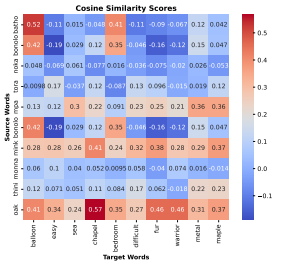


Fig. 133: lr-t and hr

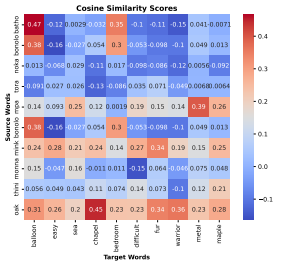


Fig. 134: lr-t and hr

Fig. 135: zul Mono Emb plots of 50 (left), 100(middle), and 200 (right), nso test

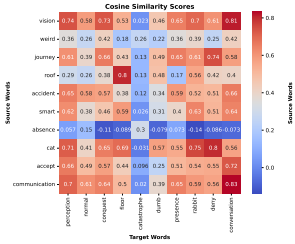


Fig. 136: hr

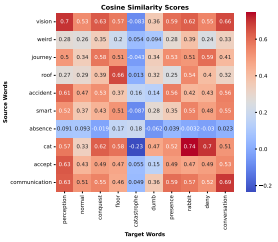


Fig. 137: hr

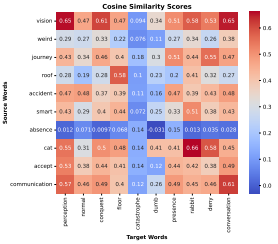


Fig. 138: hr

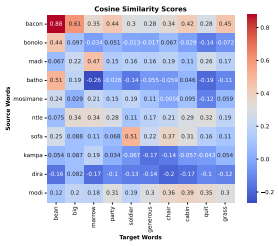


Fig. 139: lr-t and hr

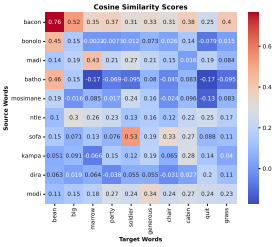


Fig. 140: lr-t and hr

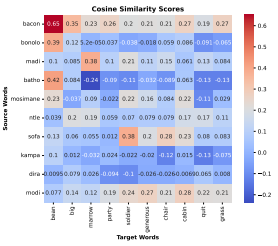


Fig. 141: lr-t and hr

Fig. 142: zul Mono plots of 50 (left), 100(middle), and 200 (right), tsn test

not the case for English Glove embeddings as reflected by high monolingual similarity scores. However, using cross-lingual embeddings derived using the three projection techniques: Canonical Correlation Analyses (CCA), VecMap, and Muse, improvements of the two similarity modes were observed. Figs. 143 to 197, represent cosine similarity plots for CCA, Figs. 199 to 253, show cosine plots for Muse embeddings, while Figs. 255 to 309, show plots for Muse projection techniques. Interestingly, closely similar word such as 'bonolo' in Setswana - translation 'easy', and the word 'simple', were captured with high scores while similarity scores of words in the monolingual case appear not to capture any meaning (e.g, sometimes very high for very distant words). This means that, in the cross-lingual case, the independently trained vector spaces are somehow projected into a common shared space where the geometrical distance of the vectors captures word similarity and meaning even for words not belonging to the same language.

B.3 Spearman's correlations

The SimLex-999 dataset, unlike other word similarity datasets, measures word similarity outside of relatedness or association. This is often measured using Spearman's correlation of the cosine similarity scores of the word pairs and the human-annotated scores. In this Appendix, we measured the Spearman's correlation of the cosine scores for our Setswana (tsn), and Sesotho (sot) monolingual embeddings (see Figs. 311 to 323 respectively). We only considered tsn, and sot embeddings since these are the two languages closest to our available test datasets of tsn and nso released by makgatho2021training. The

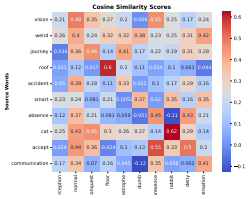


Fig. 143: hr

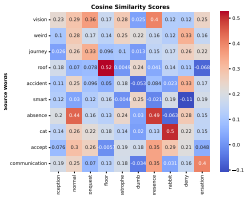


Fig. 144: hr

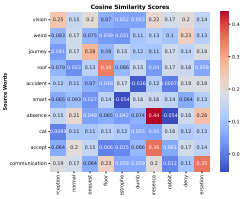


Fig. 145: hr

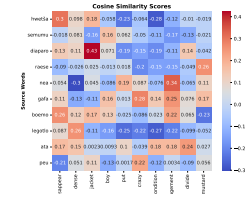


Fig. 146: lr-t and hr

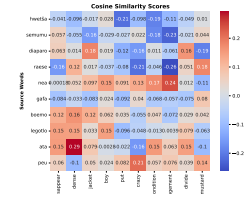


Fig. 147: lr-t and hr

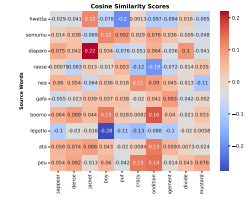


Fig. 148: lr-t and hr

Fig. 149: sot Emb CCA plots of 50 (left), 100(middle), and 200 (right), nso test

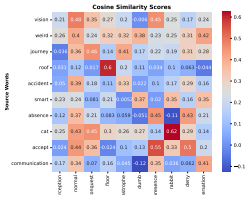


Fig. 150: hr

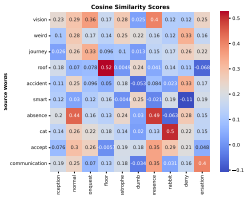


Fig. 151: hr

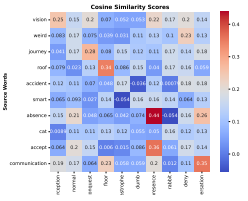


Fig. 152: hr

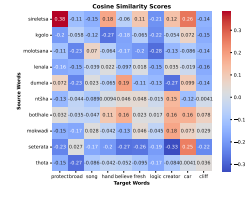


Fig. 153: lr-t and hr

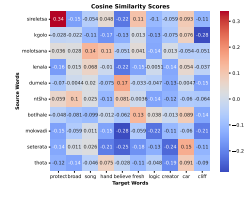


Fig. 154: lr-t and hr

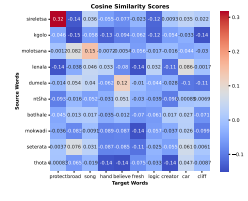


Fig. 155: lr-t and hr

Fig. 156: sot Emb CCA plots of 50 (left), 100(middle), and 200 (right), tsn test

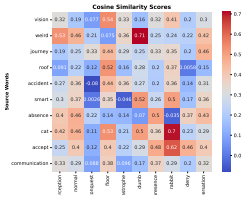


Fig. 157: hr

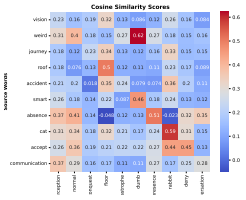


Fig. 158: hr

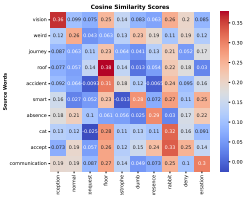


Fig. 159: hr

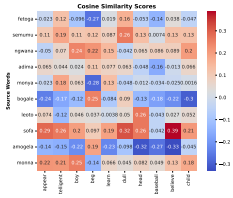


Fig. 160: lr-t and hr

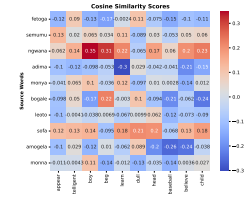


Fig. 161: lr-t and hr

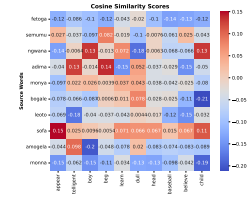


Fig. 162: lr-t and hr

Fig. 163: tsn CCA plots of 50 (left), 100(middle), and 200 (right), nso test

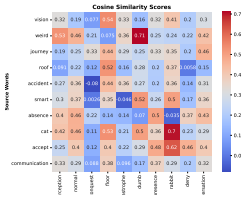


Fig. 164: hr

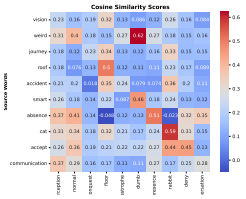


Fig. 165: hr

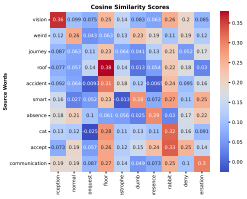


Fig. 166: hr

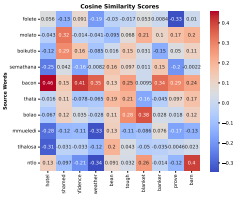


Fig. 167: lr-t and hr

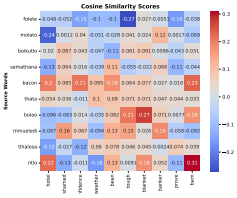


Fig. 168: lr-t and hr

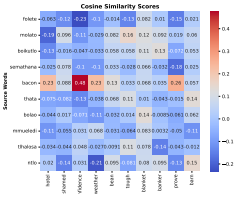


Fig. 169: lr-t and hr

Fig. 170: tsn CCA plots of 50 (left), 100(middle), and 200 (right), tsn test

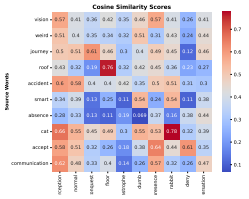


Fig. 171: hr

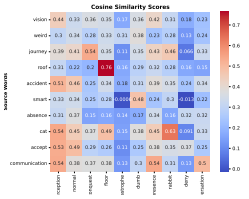


Fig. 172: hr

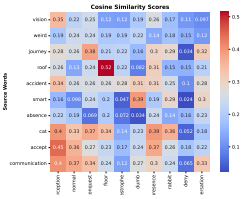


Fig. 173: hr

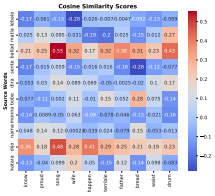


Fig. 174: lr-t and hr

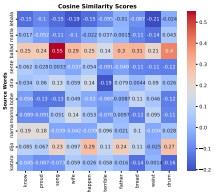


Fig. 175: lr-t and hr

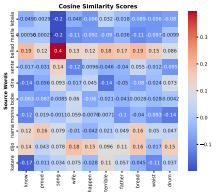


Fig. 176: lr-t and hr

Fig. 177: xho CCA plots of 50 (left), 100(middle), and 200 (right), nso test

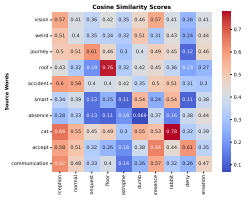


Fig. 178: hr

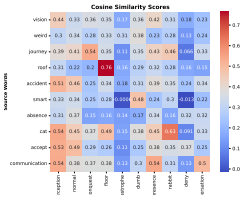


Fig. 179: hr

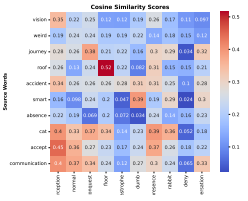


Fig. 180: hr

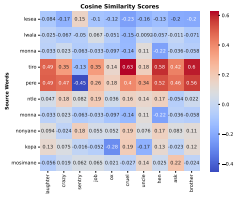


Fig. 181: lr-t and hr

Fig. 182: lr-t and hr

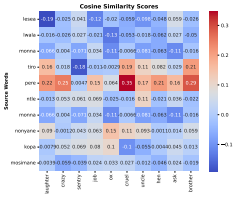


Fig. 183: lr-t and hr

Fig. 184: xho CCA plots of 50 (left), 100(middle), and 200 (right), tsn test

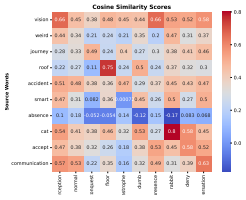


Fig. 185: hr

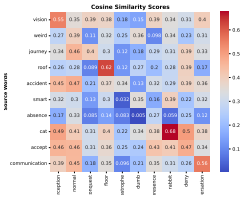


Fig. 186: hr

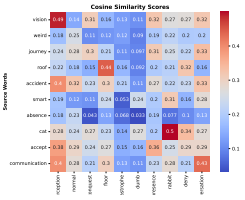


Fig. 187: hr

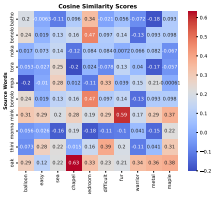


Fig. 188: lr-t and hr

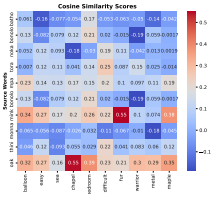


Fig. 189: lr-t and hr

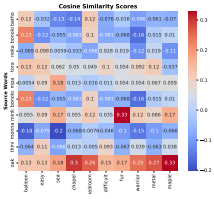


Fig. 190: lr-t and hr

Fig. 191: zul CCA plots of 50 (left), 100(middle), and 200 (right), nso test

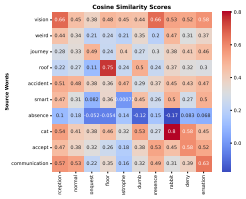


Fig. 192: hr

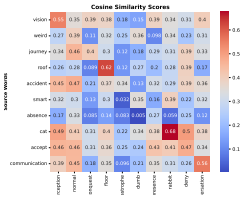


Fig. 193: hr

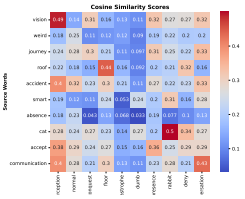


Fig. 194: hr

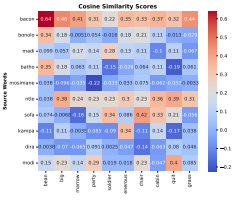


Fig. 195: lr-t and hr

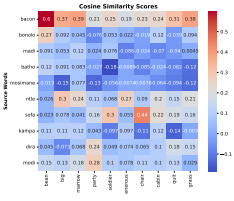


Fig. 196: lr-t and hr

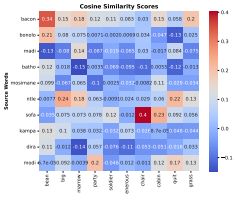


Fig. 197: lr-t and hr

Fig. 198: zul CCA plots of 50 (left), 100(middle), and 200 (right), tsn test

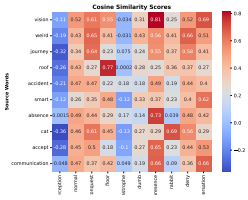


Fig. 199: hr

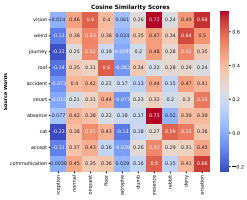


Fig. 200: hr

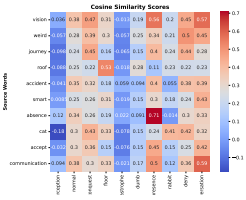


Fig. 201: hr

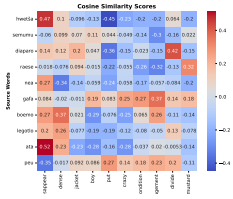


Fig. 202: lr-t and hr

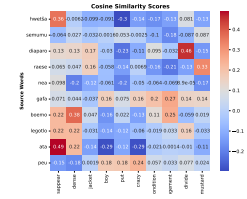


Fig. 203: lr-t and hr

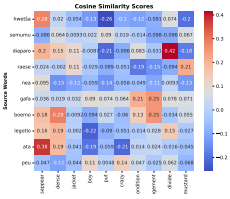


Fig. 204: lr-t and hr

Fig. 205: sot Emb Muse plots of 50 (left), 100(middle), and 200 (right), nso test

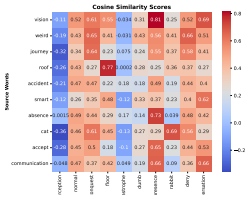


Fig. 206: hr

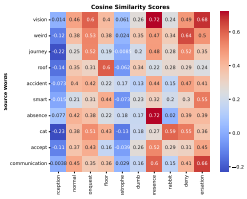


Fig. 207: hr

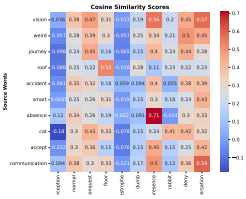


Fig. 208: hr

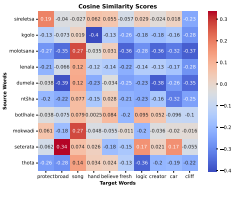


Fig. 209: lr-t and hr

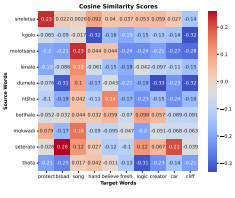


Fig. 210: lr-t and hr

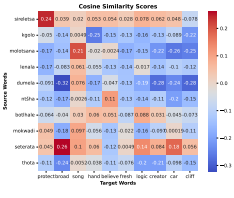


Fig. 211: lr-t and hr

Fig. 212: sot Emb Muse plots of 50 (left), 100(middle), and 200 (right), tsn test

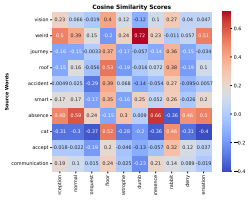


Fig. 213: hr

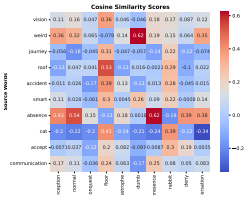


Fig. 214: hr

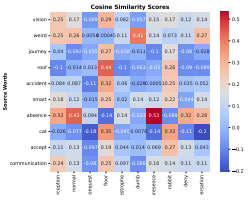


Fig. 215: hr

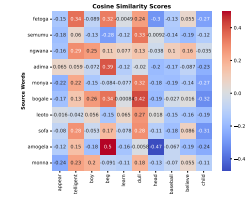


Fig. 216: lr-t and hr

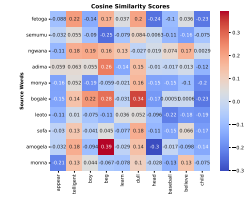


Fig. 217: lr-t and hr

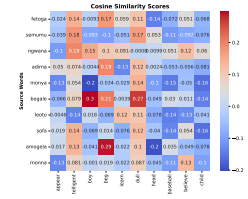


Fig. 218: lr-t and hr

Fig. 219: tsn Muse plots of 50 (left), 100(middle), and 200 (right), nso test

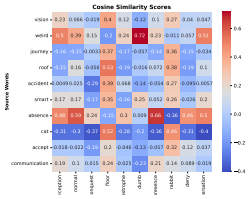


Fig. 220: hr

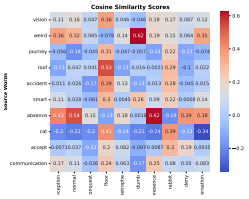


Fig. 221: hr

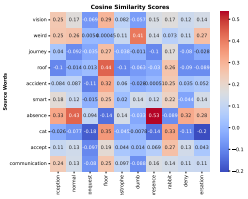


Fig. 222: hr

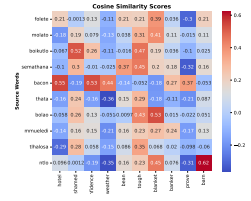


Fig. 223: lr-t and hr

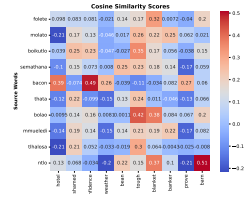


Fig. 224: lr-t and hr

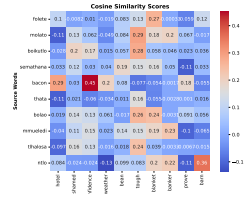


Fig. 225: lr-t and hr

Fig. 226: tsn Muse plots of 50 (left), 100(middle), and 200 (right), tsn test

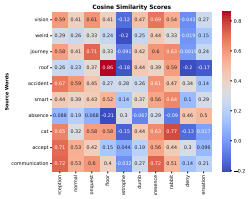


Fig. 227: hr

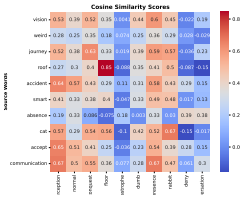


Fig. 228: hr

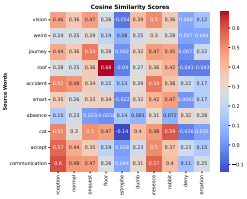


Fig. 229: hr

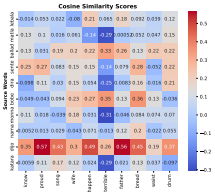


Fig. 230: lr-t and hr

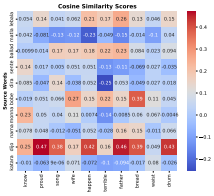


Fig. 231: lr-t and hr

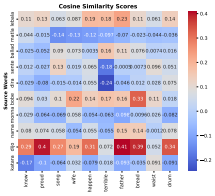


Fig. 232: lr-t and hr

Fig. 233: xho Muse plots of 50 (left), 100(middle), and 200 (right), nso test

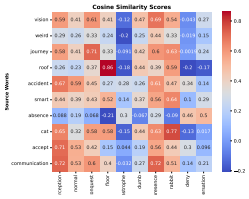


Fig. 234: hr

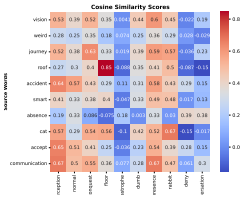


Fig. 235: hr

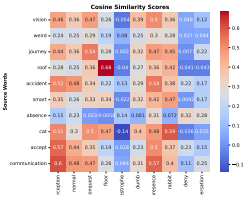


Fig. 236: hr

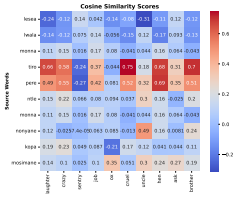


Fig. 237: lr-t and hr

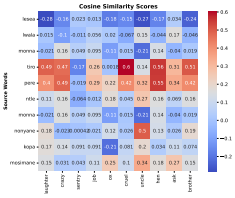


Fig. 238: lr-t and hr

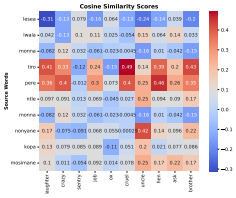


Fig. 239: lr-t and hr

Fig. 240: xho Muse plots of 50 (left), 100(middle), and 200 (right), tsn test

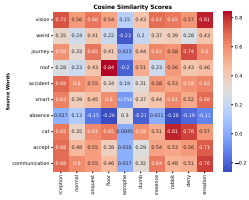


Fig. 241: hr

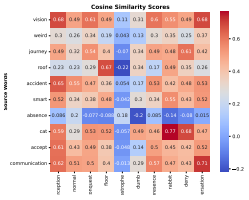


Fig. 242: hr

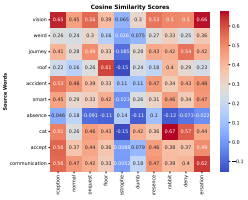


Fig. 243: hr

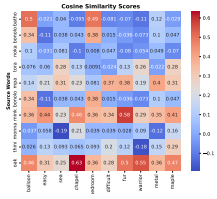


Fig. 244: lr-t and hr

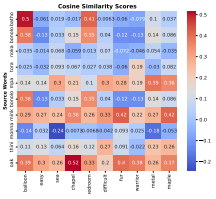


Fig. 245: lr-t and hr

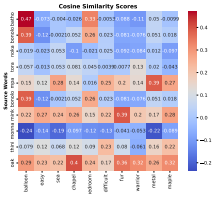


Fig. 246: lr-t and hr

Fig. 247: zul Muse plots of 50 (left), 100(middle), and 200 (right), nso test

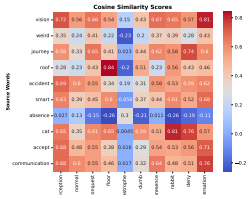


Fig. 248: hr

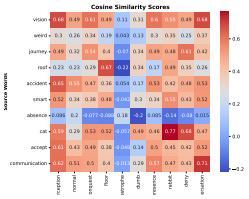


Fig. 249: hr

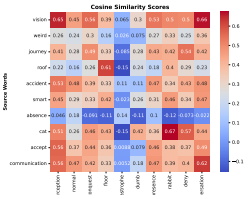


Fig. 250: hr

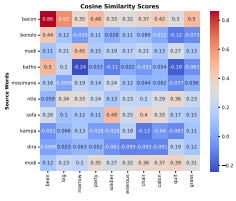


Fig. 251: lr-t and hr

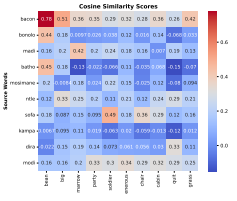


Fig. 252: lr-t and hr

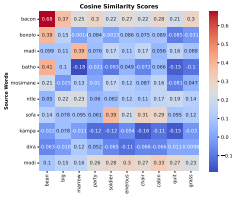


Fig. 253: lr-t and hr

Fig. 254: zul VecMap plots of 50 (left), 100(middle), and 200 (right), tsn test

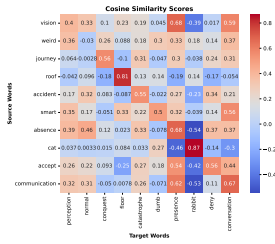


Fig. 255: hr

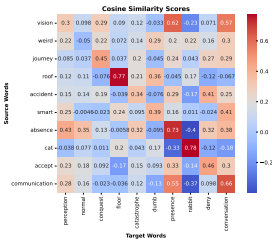


Fig. 256: hr

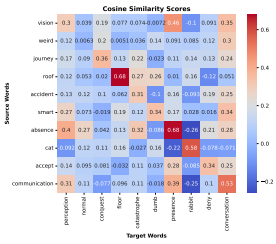


Fig. 257: hr

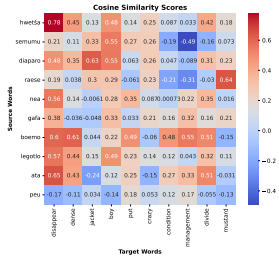


Fig. 258: lr-t and hr

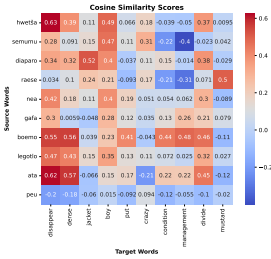


Fig. 259: lr-t and hr

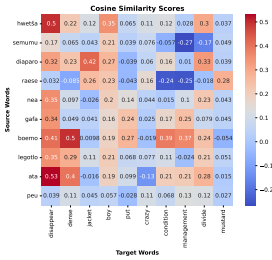


Fig. 260: lr-t and hr

Fig. 261: sot Emb VecMap plots of 50 (left), 100(middle), and 200 (right), nso test

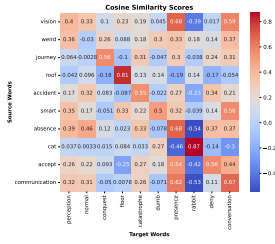


Fig. 262: hr

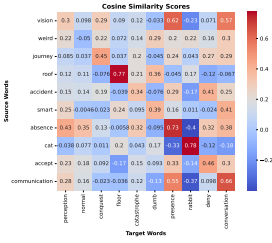


Fig. 263: hr

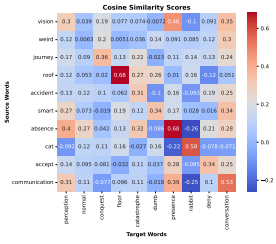


Fig. 264: hr

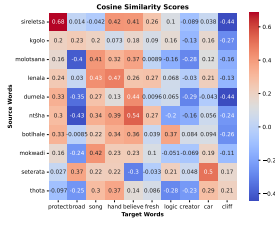


Fig. 265: lr-t and hr

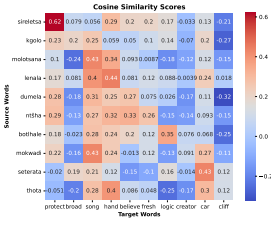


Fig. 266: lr-t and hr

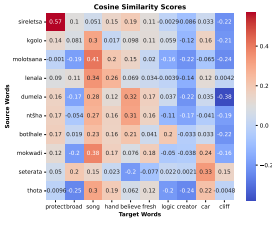


Fig. 267: lr-t and hr

Fig. 268: sot Emb VecMap plots of 50 (left), 100(middle), and 200 (right), tsn test

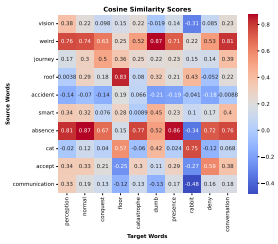


Fig. 269: hr

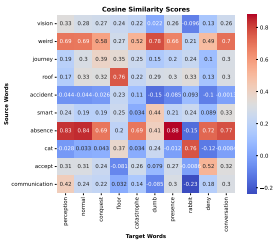


Fig. 270: hr

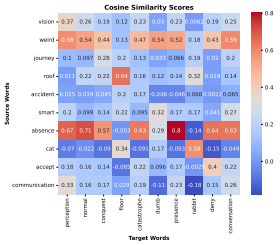


Fig. 271: hr

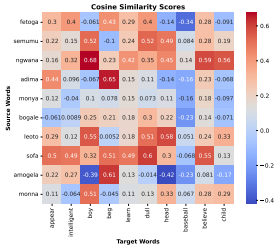


Fig. 272: lr-t and hr

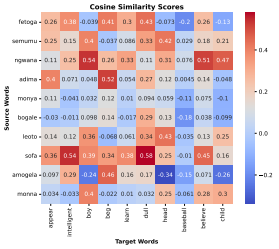


Fig. 273: lr-t and hr

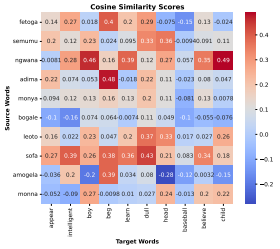


Fig. 274: lr-t and hr

Fig. 275: sot VecMap plots of 50 (left), 100(middle), and 200 (right), nso test

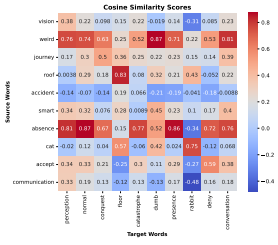


Fig. 276: hr

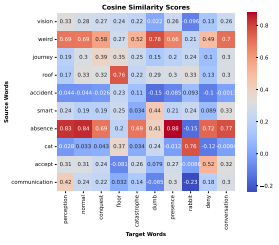


Fig. 277: hr

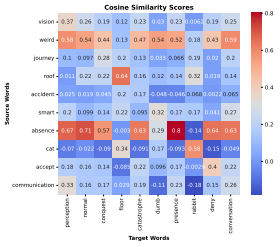


Fig. 278: hr

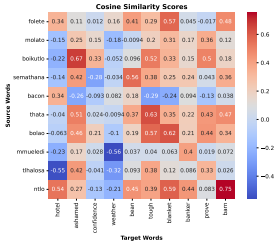


Fig. 279: lr-t and hr

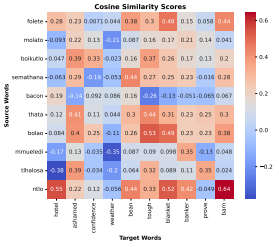


Fig. 280: lr-t and hr

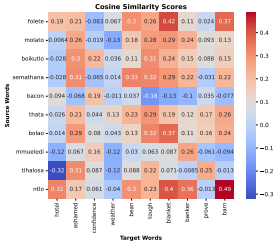


Fig. 281: lr-t and hr

Fig. 282: sot VecMap plots of 50 (left), 100(middle), and 200 (right), tsn test

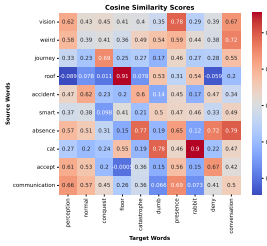


Fig. 283: hr

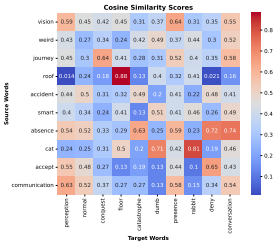


Fig. 284: hr

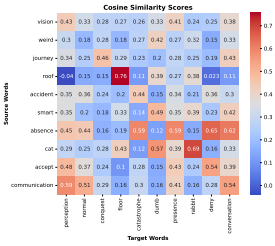


Fig. 285: hr

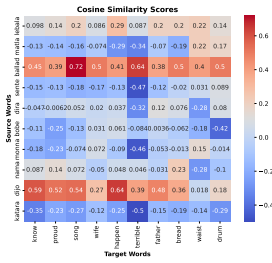


Fig. 286: lr-t and hr

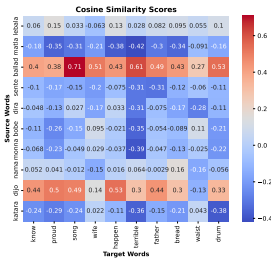


Fig. 287: lr-t and hr

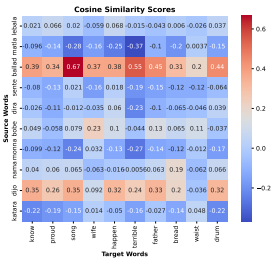


Fig. 288: lr-t and hr

Fig. 289: xho VecMap plots of 50 (left), 100(middle), and 200 (right), nso test

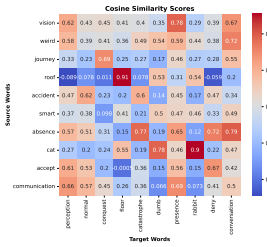


Fig. 290: hr

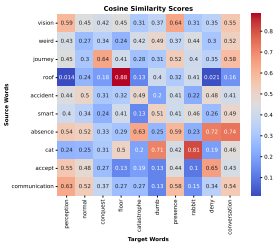


Fig. 291: hr

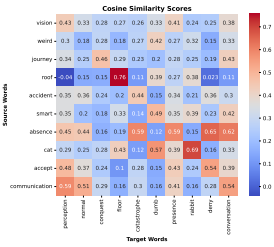


Fig. 292: hr

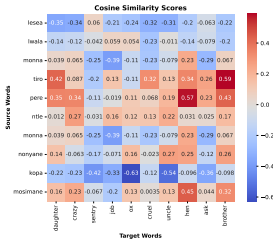


Fig. 293: lr-t and hr

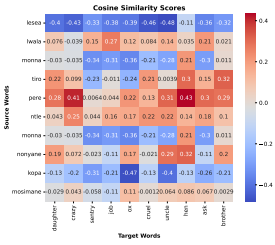


Fig. 294: lr-t and hr

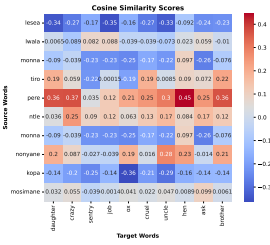


Fig. 295: lr-t and hr

Fig. 296: xho VecMap plots of 50 (left), 100(middle), and 200 (right), tsn test

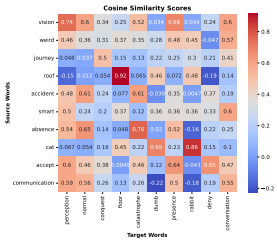


Fig. 297: hr

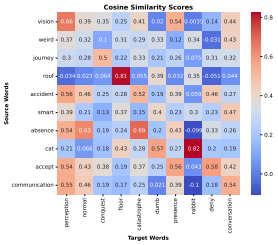


Fig. 298: hr

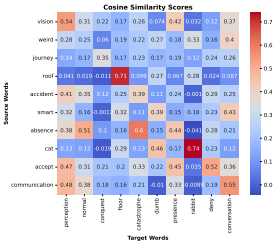


Fig. 299: hr

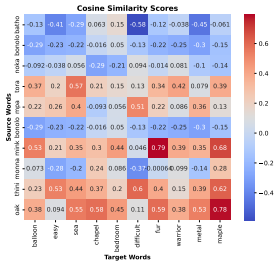


Fig. 300: lr-t and hr

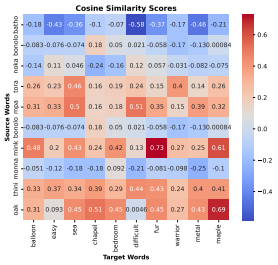


Fig. 301: lr-t and hr

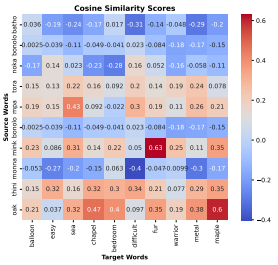


Fig. 302: lr-t and hr

Fig. 303: zul VecMap plots of 50 (left), 100(middle), and 200 (right), nso test

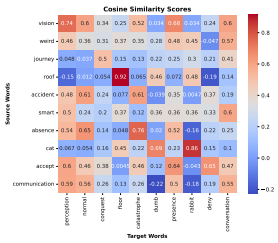


Fig. 304: hr

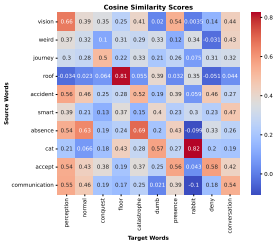


Fig. 305: hr

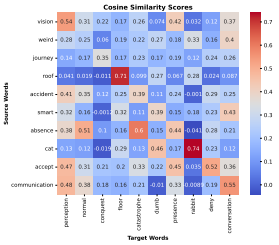


Fig. 306: hr

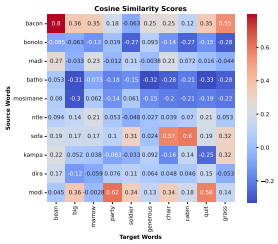


Fig. 307: lr-t and hr

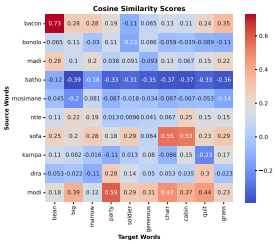


Fig. 308: lr-t and hr

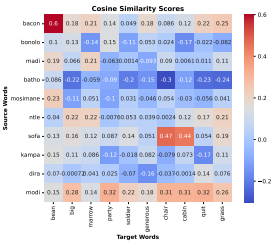


Fig. 309: lr-t and hr

Fig. 310: zul VecMap plots of 50 (left), 100(middle), and 200 (right), tsn test

Spearman scores generated using monolingual embeddings indicate low values for all datasets and all embedding dimensions. This is expected since the cosine scores are generated from independently trained English and the paired low-resourced language embeddings. Moreover, we measured the Spearman's correlation using cross-lingual embeddings generated with CCA Figs. 325 to 337, Muse Figs. 339 to 351, and VecMap Figs. 353 to 365. Unlike heatmaps in the monolingual case, cross-lingual embeddings generate spread-out heatmaps with increased scores across languages. Nevertheless, significant improvement is realized on the Sesotho (sot), and Setswana (tsn) cross-lingual embeddings when using Muse cross-lingual projections followed by VecMap embeddings. CCA embeddings generated the least performing Spearman's scores. These results are consistent with previous cosine similarity results outlining the semantic closeness of embeddings created using the Muse technique compared to CCA and VecMap embeddings. Finally, although the Spearman scores in the plots are low, they still manage to capture the catapult of moving from monolingual representation semantics to cross-lingual representation semantics, illustrating the need to further investigate cross-lingual projection techniques.

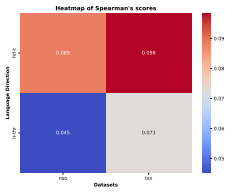


Fig. 311: dim=50

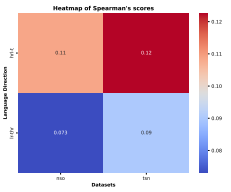


Fig. 312: dim=100

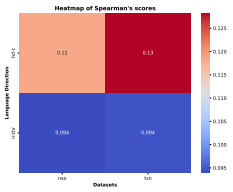


Fig. 313: dim=200

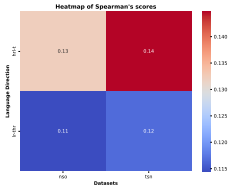


Fig. 314: dim=50

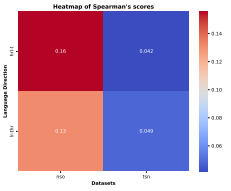


Fig. 315: dim=100

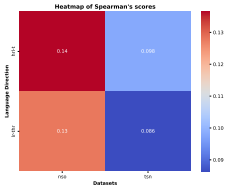


Fig. 316: dim=200

Fig. 317: Spearman's correlation: tsn Mono Emb on nso test (row 1) and tsn test (row 2)

C Cross-lingual and Monolingual Results extended

Table 7, 8, and 9, reports the Accuracy (acc), Precision (prec), and Recall (rec), of Afro-XLM-r base model trained on explicit CCA, VecMap, and Muse cross-lingual em-

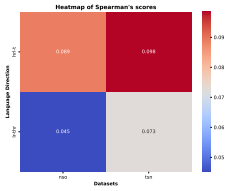


Fig. 318: dim=50

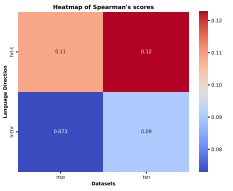


Fig. 319: dim=100

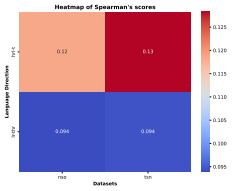


Fig. 320: dim=200

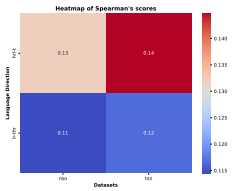


Fig. 321: dim=50

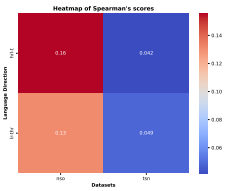


Fig. 322: dim=100

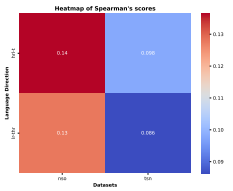


Fig. 323: dim=200

Fig. 324: Spearman's correlation: sot Mono Emb on nso test (row 1) and tsn test (row 2)

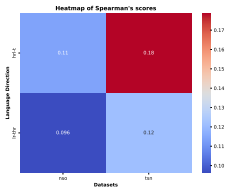


Fig. 325: dim=50

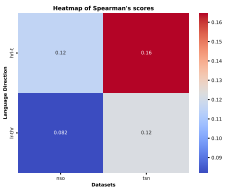


Fig. 326: dim=100

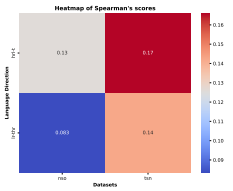


Fig. 327: dim=200

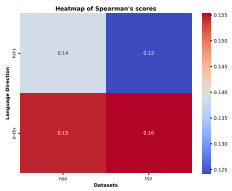


Fig. 328: dim=50

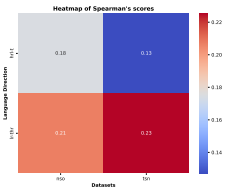


Fig. 329: dim=100

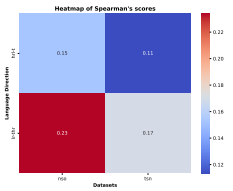


Fig. 330: dim=200

Fig. 331: Spearman's correlation: en-tsn CCA cross Emb on nso test (row 1) and tsn test (row 2)

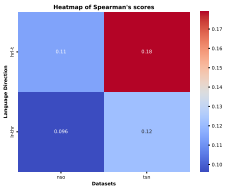


Fig. 332: dim=50

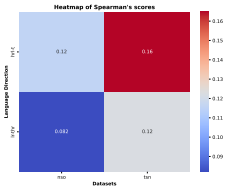


Fig. 333: dim=100

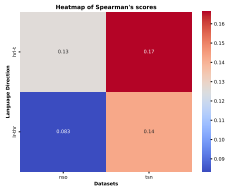


Fig. 334: dim=200

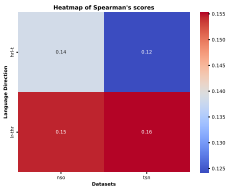


Fig. 335: dim=50

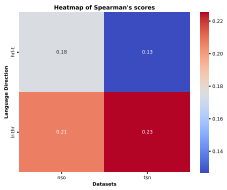


Fig. 336: dim=100

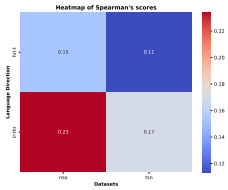


Fig. 337: dim=200

Fig. 338: Spearman's correlation: en-sot CCA cross Emb on nso test (row 1) and tsn test (row 2)

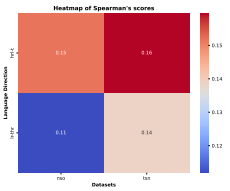


Fig. 339: dim=50

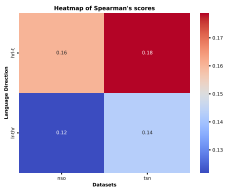


Fig. 340: dim=100

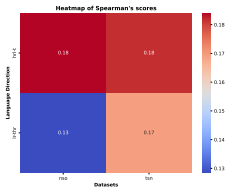


Fig. 341: dim=200

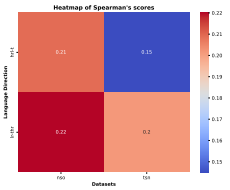


Fig. 342: dim=50

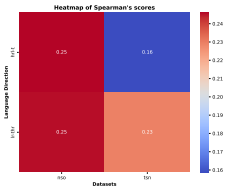


Fig. 343: dim=100

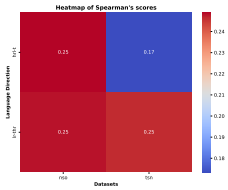


Fig. 344: dim=200

Fig. 345: Spearmans correlation: en-tsn Muse cross Emb on nso test (row 1) and tsn test (row 2)

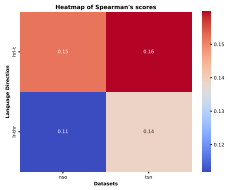


Fig. 346: dim=50

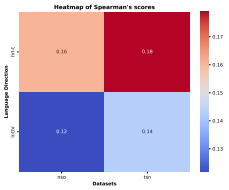


Fig. 347: dim=100

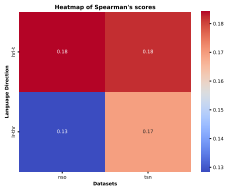


Fig. 348: dim=200

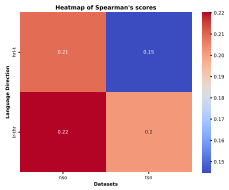


Fig. 349: dim=50

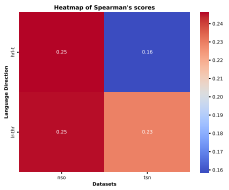


Fig. 350: dim=100

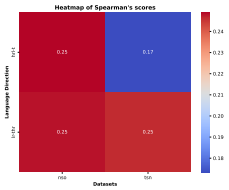


Fig. 351: dim=200

Fig. 352: Spearmans correlation: en-sot Muse cross Emb on nso test (row 1) and tsn test (row 2)

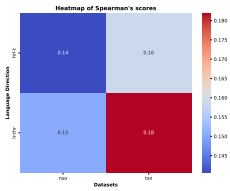


Fig. 353: dim=50

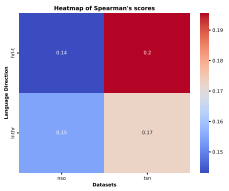


Fig. 354: dim=100

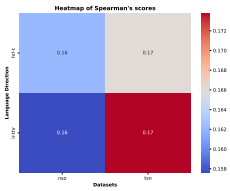


Fig. 355: dim=200

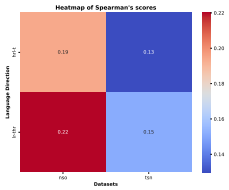


Fig. 356: dim=50

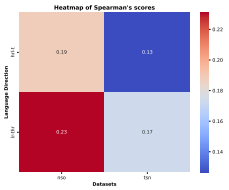


Fig. 357: dim=100

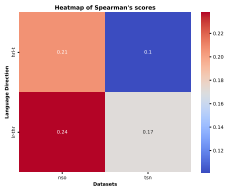


Fig. 358: dim=200

Fig. 359: Spearmans correlation: en-tsn VecMap cross Emb on nso test (row 1) and tsn test (row 2)

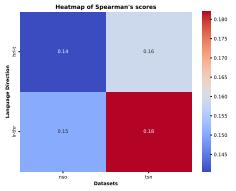


Fig. 360: dim=50

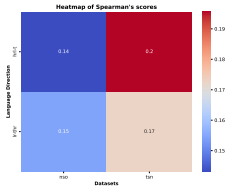


Fig. 361: dim=100

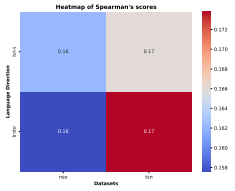


Fig. 362: dim=200

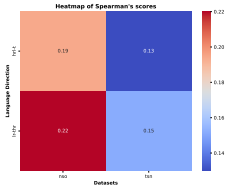


Fig. 363: dim=50

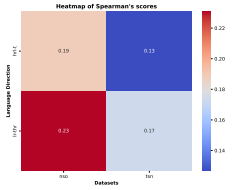


Fig. 364: dim=100

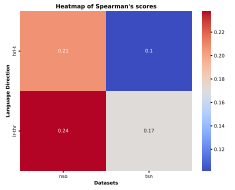


Fig. 365: dim=200

Fig. 366: Spearman's correlation: en-sot VecMap cross Emb on nso test (row 1) and tsn test (row 2)

beddings respectively. Our results indicate that the models perform well under accuracy evaluation; however, they do not do very well with other metrics such as precision and recall. This could be due to the accuracy being misleading under imbalanced classes. For Afro-XLM-r model, Muse embeddings seem to perform better on some datasets compared to the rest of the techniques. Additionally, there is no consistent pattern indicating which embedding technique or dimension performs consistently better than the rest. Similarly, Table 10, 11, and 12 show a similar trend for the Serengeti model. Comparatively, Serengeti performed better than Afro-XLM-r. We hypothesize this could stem from efficiently learned weights of the Serengeti model and the size difference, since Serengeti is significantly larger than Afro-XLMr-r. Moreover, this could be due to Serengeti being trained on more African languages compared to Afro-XLM-r, resulting in better transfer of performance gains.

Our vector matrix composing the embedding layer (i.e. the injected matrix) was created using randomly selected word vectors from the cross-lingual embedding word-vector pairs. We hypothesized that this could be improved by selecting words from the training dataset as word vectors to build the matrix. For this, Table 13, 14, and 15, show results of the injected Afro-XLM-r models created by using words from the code switch training datasets. Results show no improvements on the randomly selected word vectors. We tested the same experiments on monolingual embeddings and the results are reported in Table 16 with a similar pattern. We have no clear explanation as to why the results show no further improvements when using vectors of the training data as injection embeddings, and therefore leave this for future works.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	97	97	94	94	76	74	90	81	80	89	89
prec	81	84	80	82	59	61	75	65	67	77	78
rec	82	85	75	78	56	55	73	59	66	79	80
Embedding Dimension: 100											
acc	96	97	89	82	85	86	91	92	89	89	72
prec	80	83	79	69	70	73	76	78	79	77	59
rec	82	84	75	64	69	76	73	74	81	79	55
Embedding Dimension: 200											
acc	86	86	94	88	84	81	69	85	89	87	77
prec	73	65	80	75	69	68	53	72	79	73	63
rec	71	59	76	69	68	67	46	71	81	71	58

Table 7: Afro-XLMr-base + CCA : Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection created through random selection of words to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	95	97	97	97	93	78	95	93	80	79	63
prec	76	83	85	85	80	65	83	78	68	66	49
rec	75	85	86	83	81	65	84	75	67	61	43
Embedding Dimension: 100											
acc	97	97	95	97	85	86	95	68	88	81	79
prec	81	83	79	85	70	73	82	49	75	69	63
rec	82	85	77	84	69	76	83	44	76	68	62
Embedding Dimension: 200											
acc	86	97	96	90	90	86	94	85	89	77	89
prec	72	83	80	78	73	74	79	71	77	63	77
rec	71	85	79	74	72	77	80	69	79	57	79

Table 8: Afro-XLMr-base + VecMap: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through a random selection of words to build the vector matrix.

D Cross-lingual heatmap scores analyses

In this section, we highlight a few instances where token attention analyses highlight the consideration of external words outside the focal sentence for improved text processing. That is, certain words outside the focal sentence show high attention scores compared to words constituting the observed sentence, resulting in correct predictions. For example, Figure 367, shows the attention heatmap of the code-switched sentence 'O utlwile ore dineo o rileng itll take the pressure off' with only the highest priority score of 0.14 for local words, while the same sentence shows higher scores of 0.8 when considering a global embedding space (all embeddings) in Figure 368. This means that a richer context is gained externally from the word set of the focal sentence, arguably made

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	96	97	86	88	90	85	92	93	78	79	87
prec	75	79	65	69	69	69	71	72	59	62	67
rec	78	80	62	67	72	72	75	76	61	63	72
Embedding Dimension: 100											
acc	85	97	95	94	91	85	93	84	87	88	87
prec	68	80	75	74	70	69	72	64	68	69	67
rec	67	82	77	75	74	73	76	65	73	75	73
Embedding Dimension: 200											
acc	96	97	95	95	90	86	93	93	88	88	88
prec	74	79	75	74	70	69	72	73	69	69	69
rec	78	81	78	76	73	74	77	78	75	74	75

Table 9: Afro-XLMr-base + Muse: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through a random selection of words to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	96	97	96	96	93	86	94	95	89	89	89
prec	77	78	79	80	78	72	79	80	76	76	76
rec	81	82	82	82	80	76	81	82	79	78	79
f1	79	80	80	81	79	74	80	81	78	77	78
Embedding Dimension: 100											
acc	96	97	96	96	93	87	94	95	89	89	89
prec	77	79	78	81	78	72	80	80	76	76	77
rec	81	82	81	82	79	76	82	82	79	79	79
f1	78	80	80	82	79	74	81	81	78	77	78
Embedding Dimension: 200											
acc	96	97	96	96	93	87	94	95	89	89	90
prec	77	80	79	81	78	73	80	81	76	77	77
rec	81	83	81	82	80	77	82	83	79	80	79
f1	79	82	80	82	79	75	81	82	78	78	78

Table 10: Serengeti + CCA: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through random selection of words to build the vector matrix.

possible by the extended semantically connected vocabulary derived from the observed languages. We believe that this is made possible by the projection of the embedding space of English and Setswana into the same shared space, thus extending the context of the overall embedding space, since the observed pattern does not exist in the combined but semantically disjoint monolingual embeddings space of English (Glove) and Setswana (FastText) embeddings (see Figure 370 and 371 of local attention and global attention heatmaps respectively).

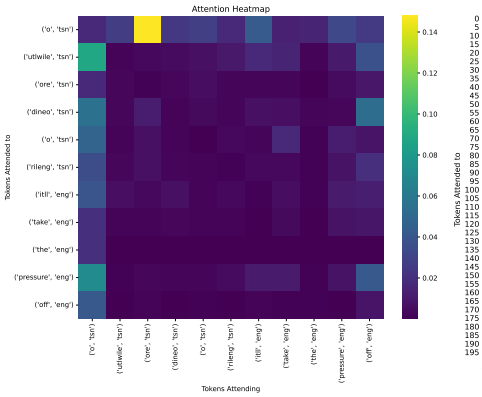


Fig. 367: VecMap local attention

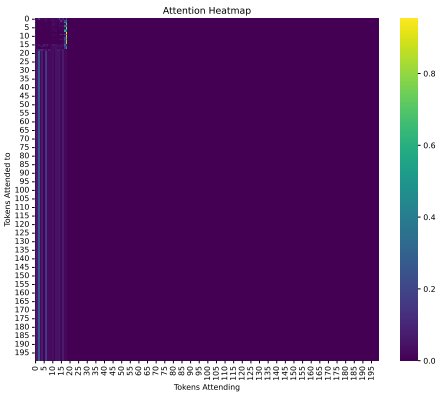


Fig. 368: VecMap global attention

Fig. 369: Embedding Attention heatmap visualization of Cross-lingual (CL) von Setswana (tsn) sentence

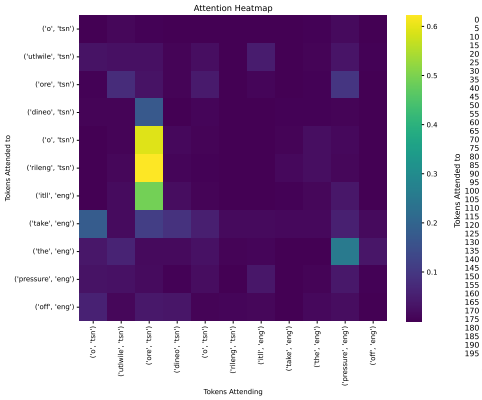


Fig. 370: Mono local attention

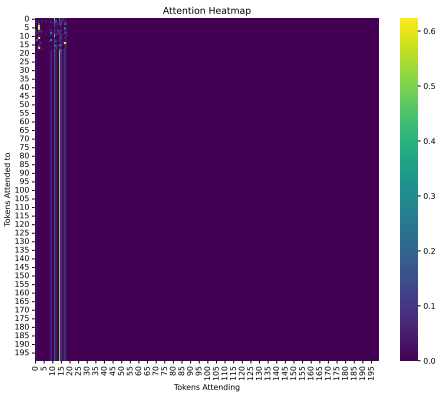


Fig. 371: Mono global attention

Fig. 372: Embedding Attention heatmap visualization of Cross-lingual (CL) von Setswana (tsn) sentence

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	96	97	96	-	93	86	94	95	89	89	89
f1	78	80	79	-	78	74	80	81	77	77	77
prec	76	79	78	-	77	72	78	80	76	76	76
rec	81	82	81	-	79	76	81	82	79	79	79
Embedding Dimension: 100											
acc	96	97	96	-	93	87	94	95	89	89	89
prec	77	79	79	-	78	72	79	81	76	76	76
rec	81	83	81	-	79	76	81	82	79	79	79
f1	79	81	80	-	79	74	80	81	77	77	78
Embedding Dimension: 200											
acc	96	97	96	-	93	87	94	95	89	89	90
prec	77	80	79	-	79	73	80	82	77	76	78
rec	81	82	81	-	80	77	82	83	79	79	80
f1	79	81	80	-	79	75	81	82	78	78	79

Table 11: Serengeti + VecMap: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through a random selection of words to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	96	96	95	95	91	86	93	94	88	88	88
prec	73	75	74	76	71	68	73	73	70	69	70
rec	79	81	78	79	75	75	78	79	75	76	75
f1	76	78	76	77	73	72	75	76	72	73	72
Embedding Dimension: 100											
acc	96	96	95	96	92	86	93	94	88	88	88
prec	73	75	74	77	73	67	73	74	69	70	70
rec	80	81	79	80	77	75	79	79	76	76	76
f1	76	78	77	78	75	71	76	77	73	73	73
Embedding Dimension: 200											
acc	96	96	95	96	92	86	93	94	88	88	88
prec	73	74	75	77	73	68	74	75	70	70	71
rec	80	81	78	80	77	75	79	80	76	76	77
f1	76	77	76	78	75	71	77	77	73	73	74

Table 12: Serengeti + Muse: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through a random selection of words to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	94	82	84	94	76	72	67	76	73	89	65
prec	76	61	66	80	58	59	51	60	58	75	51
rec	72	57	55	76	55	53	45	57	54	77	44
f1	74	58	60	78	56	55	47	58	55	76	46
Embedding Dimension: 100											
acc	94	79	81	84	67	69	62	85	71	69	72
prec	75	55	71	70	47	56	45	71	56	57	59
rec	72	45	63	60	43	52	35	69	53	48	55
f1	73	48	66	64	44	54	39	70	54	51	56
Embedding Dimension: 200											
acc	90	74	89	94	84	60	82	66	71	62	68
prec	71	50	73	80	66	49	65	49	57	50	52
rec	63	43	64	77	64	41	60	43	54	31	44
f1	66	45	67	79	65	44	62	45	55	38	47

Table 13: Afro-XLMr-base + CCA: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through word selection from training data to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	86	85	89	79	67	66	72	56	71	77	63
prec	70	64	76	65	47	53	56	37	56	65	48
rec	68	58	72	58	42	44	48	29	53	58	42
f1	69	60	73	61	44	47	51	32	54	61	45
Embedding Dimension: 100											
acc	83	97	88	82	59	64	71	80	57	74	68
prec	63	82	74	65	36	52	54	63	40	61	52
rec	58	84	70	61	29	43	47	58	29	50	44
f1	60	83	71	63	32	46	49	60	33	55	47
Embedding Dimension: 200											
acc	93	74	81	88	75	75	66	85	65	78	69
prec	70	50	71	74	56	59	50	69	50	64	50
rec	65	42	62	68	53	55	44	68	46	58	42
f1	67	45	65	71	54	57	46	68	47	60	45

Table 14: Afro-XLMr-base + VecMap: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through word selection from training data to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 50											
acc	88	86	91	82	88	85	88	76	78	79	79
prec	63	61	64	62	63	65	65	54	58	59	60
rec	49	55	61	57	66	70	65	52	60	61	61
f1	54	57	62	60	65	68	65	52	58	60	60
Embedding Dimension: 100											
acc	80	73	91	81	81	85	83	91	61	84	78
prec	59	47	67	62	57	68	61	67	42	64	58
rec	46	39	64	58	58	72	51	69	38	64	59
f1	51	42	65	59	57	70	54	68	39	64	58
Embedding Dimension: 200											
acc	82	85	73	79	90	68	73	83	61	79	66
prec	59	57	60	60	69	54	53	62	42	62	45
rec	51	52	48	55	73	51	42	63	38	63	38
f1	54	54	52	57	71	51	46	62	39	62	41

Table 15: Afro-XLMr-base + Muse: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through word selection from training data to build the vector matrix.

	sot	tsn	zul	xho	zulxho	sottsn	zulxhosot	zulxhotsn	sottsnzul	sottsnxho	zulxhosottsn
Embedding Dimension: 100											
acc	93	87	72	79	81	75	81	74	78	76	75
prec	63	58	57	61	58	55	55	52	59	59	55
rec	64	56	44	54	60	55	49	51	61	55	52
f1	63	56	49	57	59	55	52	51	60	56	52
Embedding Dimension: 200											
acc	92	81	76	91	72	82	65	65	55	81	75
prec	66	56	56	72	46	63	45	44	37	61	56
rec	64	52	33	66	44	62	41	40	28	63	54
f1	65	53	41	69	45	61	42	41	31	62	54
Embedding Dimension: 50											
acc	93	81	91	91	80	82	87	83	67	76	79
prec	71	56	61	69	56	64	63	61	47	57	59
rec	68	53	62	65	56	62	64	61	40	54	61
f1	69	54	61	67	56	62	64	61	43	55	60

Table 16: Afro-XLMr-base + Monolingual: Average metrics for each embedding dimension across folders. Averaged over 5 runs. The word embeddings for injection are created through a random selection of words to build the vector matrix.