Code-switching as a cross-lingual Training Signal: an Example with Unsupervised Bilingual Embedding

Félix Gaschi [2,3], **Ilias El Baamrani**[1], **Barbara Gendron**[1], **Parisa Rastin** [2], **Yannick Toussaint** [2]
[1]École des Mines de Nancy, [2]LORIA, [3] SAS Posos
{felix.gaschi,parisa.rastin,yannick.toussaint}@loria.fr

Abstract

Code-switching is the occurrence of words from different languages in the same utterance. This paper shows that code-switching is largely present in a popular dataset for training word embeddings, and demonstrates that it can be a useful training signal for unsupervised cross-lingual embeddings. `CoSwitchMap`, the proposed method for leveraging this signal, outperforms other unsupervised mapping-based methods for cross-lingual embeddings on two of the three tested language pairs and suggests that code-switching can be a useful training signal for multilingual representations.

1 Introduction

Code-switching occurs when words from multiple languages are used in a single sentence. Some examples of code-switching, randomly sampled from a Wikipedia dump, are shown in Figure 1. While code-switching can be expected in speech data, or in informal writing, this paper shows that it can be found in more formally written data like in Wikipedia, in an amount that is sufficient to use as a training signal for learning fair multilingual representations.

While artificially induced code-switching was already shown to help build cross-lingual embeddings (Xiao and Guo, 2014; Gouws and Søgaard, 2015), this paper investigates whether it is possible to leverage naturally-occurring code-switching for the same objective.

To demonstrate the usefulness of code-switching, this work builds cross-lingual word embeddings using code-switching as a training signal. But rather than the proposed method itself, we believe that the most important part of our contribution is to show that code-switching is present in sufficient amount in a typical monolingual pre-training dataset that it can be used as a cross-lingual training signal, with our proposed method or with another.

Exemple 1 : 1999年歐洲歌唱大賽(eurovision song contest 1999) 為歐洲歌唱大賽之第44屆比賽
Exemple 2 : as a result , ” li ” (禮) , meaning ” ritual ” or ” etiquette , ”governed the conduct of the nobles , whilst ” xing ” (刑) , the rules of punishment
Exemple 3 : 是一款由鬼游(ghost town games) 公司, team 17 行的烹模游. 玩家通多人合作或多角控制, 控制多游角色挑各种房里的机

Figure 1: Examples of code-switching

The experiments in this paper focus on static word embeddings built with FastText (Bojanowski et al., 2016) rather than contextualized ones, obtained with deeper models such as BERT (Devlin et al., 2019). Static embeddings are preferred in this work for their simplicity and because there is already a whole line of work for creating cross-lingual static embeddings (Mikolov et al., 2013a; Conneau et al., 2017, inter alia), whereas pre-training contextualized embeddings require more resource and methods for improving their multilingual properties might not be consistently effective (Wu and Dredze, 2020).

There are several methods to obtain cross-lingual static embeddings. Mikolov et al. (2013a) introduce one of the pioneering methods for supervised alignment that consists of learning a mapping between the source and target language. Following the observation that word translations tend to have similar geometric properties, they leverage parallel data through a bilingual dictionary to learn a linear projection between the two languages. Even if such an approach proved efficient, it still has the drawback of being supervised. This has motivated the emergence of less supervised or even completely unsupervised alignment methods as developed in Conneau et al. (2017). Leveraging isomorphic properties between embedding spaces,

Proceedings of the The 3rd Workshop on Multi-lingual Representation Learning (MRL), pages 208–217

they describe a method to deduce a bilingual dictionary that provides an accurate word alignment and matches supervised baselines. However, fully unsupervised approaches may not be stable enough as pointed out by Søgaard et al. (2018a). They claim that the reason why unsupervised alignment can sometimes lead to lower performances is that the original embedding spaces are not really isomorphic. In addition, they showed that retrieving identical words in order to form a seed dictionary brings a weak supervision signal which is enough to improve the robustness of the approach.

To evaluate the potential of code-switching as a cross-lingual signal, this work first provides quantitative insights about the presence of code-switching in Wikipedia, showing that it covers a large part of the most frequent words of the studied languages. Thus, this paper proposes `CoSwitchMap` (**Co**de-**Switch**ing-based bilingual **Map**ping), which uses code-switching as a weak supervision signal to learn bilingual word embeddings for languages in different scripts. `CoSwitchMap` allows to overcome some known limitations of unsupervised mapping-based methods for learning multilingual word embeddings.

2 Related Works

In the following, code-switching will be referred to as the use of words from multiple languages in a single sentence or discourse. This is different from language contamination, which simply refers to the presence of whole sentences from other languages in a supposedly monolingual corpus. With code-switching, two words from different languages can share the same context, contrary to language contamination. According to Blevins and Zettlemoyer (2022), language contamination is almost surely found in large English corpora, and it might explain the cross-lingual transfer abilities of monolingual models. Indeed, even with less than 1% of contamination, supposedly monolingual models based on Transformers (Vaswani et al., 2017) such as BERT (Devlin et al., 2019) or RoBERTa (Liu et al., 2019) reach surprising performances on target languages which are positively correlated to the amount of contaminated data on POS tagging task.

Because token contamination does not combine different languages in the same context, only code-switching is studied in this work. Artificially adding some code-switching is a way to create cross-lingual embeddings. Several approaches were developed in that sense (Xiao and Guo, 2014; Gouws and Søgaard, 2015). They all have in common that some tokens are randomly replaced with their translation in monolingual training data, ensuring that translation pairs keep having the same embedding representation. According to Ruder (2017), pseudo-bilingual corpora and bilingual mapping methods are in fact equivalent because they boil down to optimizing the same objective.

On the other hand, code-switching can improve the pre-training of deep multilingual models. In order to improve the learning of contextual information mostly in mBERT, the multilingual version of BERT, Qin et al. (2020) developed a data augmentation approach by generating sentences with randomly chosen code-switched tokens. This method, used during the fine-tuning step, systematically improves the performances of baseline models on all five tasks and for each pair of languages. With the same goal of achieving language neutrality, Krishnan et al. (2021) also leverage multilingual code-switching within some model training. The main contribution of such methods is to be able to perform cross-lingual generalization with a reasonable amount of parallel data from different languages. The cross-lingual signal used for the cited methods is indeed smaller that the pre-training corpus of mBERT. A similar approach proposed by Yang et al. (2020) outperforms existing Transformer-based models with an enhanced version of the Masked Language Modeling (MLM) task performed during mBERT pre-training. By training on code-switched sentences, the model is expected to learn a cross-lingual embedding.

The previously mentioned methods focus either on multilingual models to improve their cross-lingual generalization or on alignment methods using artificially created code-switching. In this work, the aim is to leverage the code-switching naturally present in a corpus, in order to train alignment methods on static embeddings without any supervised cross-lingual signal. To the best of our knowledge, there isn't any existing method that relies on naturally occurring code-switching to produce multilingual static embeddings.

3 Method

Our goal is (1) to identify code-switching situations in monolingual corpora like Wikipedia, (2) to learn an orthogonal mapping between two monolingual embeddings by applying a modified skip-gram loss

to pairs of code-switched words, and (3) to refine this orthogonal mapping with self-learning.

3.1 Identifying code-switching with different scripts

To identify code-switching situations we must find paragraphs that contain words coming from different languages. However, determining whether a word belongs to the vocabulary of one given language is not straightforward. Without resorting to additional resources like a dictionary, the vocabulary of one language can be obtained based on occurrences in a monolingual corpus. However, if this monolingual corpus potentially contains code-switching, the vocabulary we would obtain might not help identify code-switching situations as it might include words from other languages.

If two languages are written using different scripts, most code-switching situations can be extracted by identifying paragraphs where the two scripts occur, using regular expressions with relevant character ranges. This method has, by design, a high recall, as it should only miss some situations where the word from one language is transcribed into the script of the other, which can still be seen as code-switching, or rather script-switching, situations. However, it can lack precision in some cases, because the same script can be used in different languages. For example, when extracting pairs of code-switched words involving English in a Chinese corpus, we might also retrieve German-Chinese pairs.

In our experiments, this code-switching extraction method allows us to obtain pairs of code-switched words to use as a weak supervision signal for `CoSwitchMap`.

3.2 Code-switching pairs as a supervision signal

We refer to code-switching pairs as pairs of words from two different languages present in the same context. The goal is to leverage these pairs as a multilingual signal to learn a mapping matrix W that allows us to project the words of a source language (src) to the target language (tgt). It must be noted that multi-word expressions are not getting a particular treatment, like in most word embedding algorithms. Code-switching pairs are pairs of words from different scripts found in the same sliding window of context. A multi-word expression like "Eurovision Song Contest" (cf. Figure 1) is broken down and each word that composes it will appear individually in pairs with Chinese neighboring words.

Given two monolingual embeddings for source and target languages obtained with skip-gram (Mikolov et al., 2013b) or a variant like FastText (Bojanowski et al., 2016), we can retrieve two embedding matrices for each language: the central embedding of each word x_i, i.e. the embedding that is usually used in downstream application, and the context embedding $\tilde{x}_j$, used to embed context words in the skip-gram algorithm. The goal is to continue the training of skip-gram with code-switched words in order to learn a matrix W mapping the source embedding x_i^{src} to the target embedding x_j^{tgt}. During the training, the W matrix will be either applied to the context word or central word depending on the training pair. Thus, we freeze the embedding matrices and initialize W with the identity matrix before training it.

The original monolingual skip-gram loss from (Mikolov et al., 2013b) is the following :

$$L = -\frac{1}{|C|} \sum_{w_i \in C} \sum_{w_j \in \mathcal{N}(w_i)} \log P(w_j|w_i) \qquad (1)$$

Where C is the corpus, w_i is a central word from the corpus, and w_j is a word found in $\mathcal{N}(w_i)$, the context window of the central word. $P(w_j|w_i)$ is computed with negative sampling as :

$$\log P(w_j|w_i) = \log \sigma(\tilde{x}_j^\top x_i) + \sum_{w_k \sim P_V}^{n} \log \sigma(-\tilde{x}_k^\top x_i) \qquad (2)$$

x_i is the embedding of w_i and $\tilde{x}_j$ is the context embedding of w_j. n negative examples of context words w_k are sampled randomly from a distribution P over the vocabulary V. Minimizing L in Equation 1 is maximizing the similarity of x_i with $\tilde{x}_j$ with respect to the similarity of x_i with any other randomly sampled word.

`CoSwitchMap` learns the mapping matrix W with a similar negative sampling loss, but replaces the source word embedding, either central or context, by their projection with W. The initial embedding obtained with skip-gram applied to monolingual corpora is frozen and the modified skip-gram loss is only computed for pairs of code-switched words. For a code-switching pair $(w_i^{\text{src}}, w_j^{\text{tgt}})$, where the central word w_i^{src} is in the source language script, and w_j^{tgt} is a context word in the target language

script, The goal is to project w_i^{src} to the target language. The probability $P(w_j^{\text{tgt}}|w_i^{\text{src}})$ becomes:

$$\log P(w_j^{\text{tgt}}|w_i^{\text{src}}) = \log \sigma(\tilde{x}_j^{\text{tgt}\top} W x_i^{\text{src}}) + \sum_{w_k^{\text{tgt}} \sim \mathcal{U}_{V_{\text{tgt}}}}^{n} \log \sigma(-\tilde{x}_k^{\text{tgt}\top} W x_i^{\text{src}}) \quad (3)$$

For the reversed case, where a code-Switching pair $(w_i^{\text{tgt}}, w_j^{\text{src}})$ is given, the central word is in the target language, and the context word in the source language. The mapping matrix must then be applied to the context embedding:

$$\log P(w_j^{\text{src}}|w_i^{\text{tgt}}) = \log \sigma(\tilde{x}_j^{\text{src}\top} W^{\top} x_i^{\text{tgt}}) + \sum_{w_k^{\text{src}} \sim \mathcal{U}_{V_{\text{src}}}}^{n} \log \sigma(-\tilde{x}_k^{\text{src}\top} W^{\top} x_i^{\text{tgt}}) \quad (4)$$

By enforcing the orthogonality of W, applying it to the source context embedding is actually equivalent to applying its inverse to the source central embedding. Using an orthogonal matrix also allows to preserve the distance between words from the source language. Thus, during the training steps, we orthogonalize the mapping matrix W after each update of the loss of a training batch as it was done in Conneau et al. (2017):

$$W \leftarrow (1+\beta)W - \beta(WW^T)W \quad (5)$$

Where β is a hyper-parameter, fixed to 0.01 following (Conneau et al., 2017).

3.3 Self-learning

The method from the previous section learns a mapping between two languages which might need some refining as it is obtained from noisy data. Indeed, as mentioned in Section 3.1, the unsupervised extraction of code-switching pairs can produce some unwanted pairs between other languages using the same script. `CoSwitchMap` thus involves an additional refinement step using self-learning as in many other existing unsupervised mapping-based methods.

For the proposed method, the self-learning procedure of VecMap (Artetxe et al., 2018b) is used, allowing for a controlled comparison with different kinds of initialization. The principle of this self-learning loop is to improve the alignment by iteratively learning a new bilingual dictionary from the previously learned mapping, and then a new mapping from this bilingual dictionary, and so on. In VecMap, each new dictionary is obtained with a nearest-neighbor search, and each new mapping with Procrustes (Artetxe et al., 2018b).

The self-learning procedure needs a seed dictionary to start. `CoSwitchMap` uses the same nearest-neighbor search as in the further steps of VecMap to calculate a new bilingual dictionary from the W mapping learned with code-switched pairs. The obtained dictionary can then be used as the first dictionary of the self-learning procedure of VecMap.

4 Experimental details

Our experiments are performed in three pairs of languages (English-Arabic, English-Russian, and English-Chinese) and based on tokenized Wikipedia dumps. We use FastText (Bojanowski et al., 2016) monolingual embeddings[1] and keep only the 200,000 most frequent words.

4.1 Code-switched pairs extraction

`CoSwitchMap` considers a word to belong to a given language if all its characters are in the character range of the relevant script. Character ranges for each language can be found in Appendix A.

For each non-English language (Arabic, Russian, and Chinese), code-switched pairs of words involving the non-English language and English are extracted from the non-English corpus and the English one. The pairs retained are all pairs of words in the same context, such that one matches one script and the other matches the other script. Two words are considered to be in the same context if they are in the same window of width 5, to match the default window size of the monolingual embedding we use.

pair	number
en-ar	7,848,024
en-ru	50,182,802
en-zh	23,097,625

Table 1: Number of code-switching pairs extracted

The total number of pairs for each language pair is reported in Table 1.

4.2 Learning the W mapping

Word embedding and context embedding matrices are obtained from already pre-trained FastText

[1] https://fasttext.cc/docs/en/pretrained-vectors.html

monolingual embeddings. The embedding matrices are l2-normalized and frozen while only the mapping is trained.

In each epoch, each pair of words is passed twice, with the English word as the central word and with the non-English word as the central word. Five negative samples are drawn uniformly from the context word language vocabulary, although limited to the 200,000 most frequent words since we filtered our monolingual embeddings.

The mapping is trained for five epochs with SGD optimizer, learning rate 0.1, momentum 0.9, and a batch size of 1024 pairs (including negative samples). The orthogonalization step is applied after each batch with $\beta = 0.01$ (cf. Equation 5).

4.3 Inference of the first dictionary

From the embeddings, roughly aligned with W, we obtain a seed dictionary with a nearest-neighbor search. For each word in the English vocabulary, we retrieve its nearest neighbor in the non-English embedding. Following Artetxe et al. (2018b), we also retrieve the nearest neighbor in the English embedding for each word in the non-English one.

The retrieval criterion is the Cross-domain Similarity with Local Scaling (CSLS) (Joulin et al., 2018), a modified cosine similarity that mitigates the effects of hubs, which are words that are nearest neighbor of many others. This criterion has a hyper-parameter which is the number of neighbors to include in the computation to mitigate the cosine similarity. We use 10 following Artetxe et al. (2018b).

4.4 Self-learning

For the self-learning iteration, we use the VecMap algorithm (Artetxe et al., 2018b)[2]. We simply replace the initialization with ours. All parameters are left with default values.

4.5 Evaluation

Following previous work, the aligned embeddings obtained are evaluated with Bilingual Lexicon Induction (BLI). Given a bilingual dictionary containing pairs of English words with their translation in a given language, we evaluate the top-1 accuracy of a nearest-neighbor search to retrieve the translation of a given word. We use the same CSLS criterion as before and as in VecMap for the nearest-neighbor search.

[2] https://github.com/artetxem/vecmap

The dictionaries used for evaluation are the evaluation dictionaries containing 1500 distinct words provided by Conneau et al. (2017)[3].

5 Results

Results show that (1) despite being infrequent, code-switching in a large unlabelled non-English corpus involves a large majority of the most frequent words of an English dictionary and that (2) `CoSwitchMap` provides a higher accuracy in bilingual lexicon induction that other unsupervised isometric mapping-based methods.

5.1 Amount of code-switching in text corpora

To evaluate the amount of code-switching in a corpus, we must rely on a dictionary or rather a list of words that are guaranteed to originate from a given language. Indeed if we rely only on different scripts, as in `CoSwitchMap`, we might have an issue with the precision of the code-switching retrieval as the same script can be used in different languages. Using a dictionary can lack a bit of recall, as a dictionary can hardly contain all the vocabulary used in English with all their inflections. But if the dictionary is comprehensive enough, it should provide a good lower bound of the number of code-switching situations.

We use the `3of6game` dictionary from the 6th version of the `12dicts`[4]. This dictionary contains 64,662 words. It was chosen because it is said to be oriented towards common words and was manually checked for errors, which should reduce the chance of the dictionary itself being polluted by code-switching. It is obtained from 6 advanced learners' ESL dictionaries, and contains American and British English, with inflections and neologisms.

We differentiate between token contamination and code-switching. Token contamination is simply the fact of finding an English token in a non-English corpus, but is not necessarily a code-switching situation, where the English word must be found in the same context as a non-English word (identified with its script). A code-switching situation is thus also a token contamination situation. But the reciprocal is not necessarily true.

Table 2 shows that code-switching is present in all the tested datasets. From around 500,000 situations in Arabic to more than 4 million in Russian.

[3] https://github.com/facebookresearch/MUSE

[4] http://wordlist.aspell.net/12dicts-readme

lang	tokens	token contamination			code-switching		
		coverage (%)	count	count digits	coverage (%)	count	count digits
ar	229M	44.9	1,043,396	6,511,347	38.0	486,764	6,360,450
ru	685M	55.1	5,237,773	26,063,394	50.7	4,158,232	25,637,900
zh	319M	47.6	1,720,247	3,220,332	39.4	1,174,912	3,117,309

Table 2: Presence of words from an English dictionary in three non-English Wikipedia dumps. Contamination considers all words that were found in the corpus, and code-switching considers them only if they are in the vicinity of a word written in the non-English script. "coverage" is the proportion of the dictionary that was found and "count", the number of single occurrences. The occurrences of digit tokens are given for comparison.

ranks	ar	ru	zh
1-10	100.0	100.0	100.0
11-100	100.0	100.0	100.0
101-1,000	99.3	99.1	99.7
1,001-10,000	86.8	93.2	90.0
$\geq$10,001	30.9	45.8	32.1

Table 3: Proportion (in %) of English words in a dictionary covered by code-switching situations, split by buckets of frequency rank. e.g. line "1-10" indicates the proportion of the ten most frequent words in the dictionary that are covered by code-switching situations in each non-English language.

This is a small fraction of the hundreds of millions of tokens present in each corpus. But, to comprehend what the frequency of code-switching represents, Table 2 shows that code-switching is 3 to 15 times rarer than digits. This goes on to show that code-switching is not an exceptional occurrence in a monolingual corpus like Wikipedia.

While code-switching is relatively scarce, it however covers an important portion of the English vocabulary. Indeed, Table 2 shows that code-switching situations cover up to half of the English dictionary. A breakdown by frequency shows that the most frequent words are almost all involved in code-switching situations, as shown in Table 3.

Figure 2 compares the frequency of English words in the English corpus with their frequency in a non-English corpus. It shows that the frequency of a code-switched word rarely exceeds 10^{-4}, with frequent words in English being generally more frequently code-switched than infrequent ones. While code-switching occurs mainly for the most frequent words, Table 2 shows that it covers a high proportion of the 10,000 most frequent words, which is comparable with the number of words kept for learning alignment in several unsupervised alignment methods. VecMap, for example, learns its mapping on the 20,000 most frequent words of both languages involved. This advocates for learning an orthogonal mapping based on code-switching pairs, as proposed in `CoSwitchMap` rather than learning entirely new embeddings.

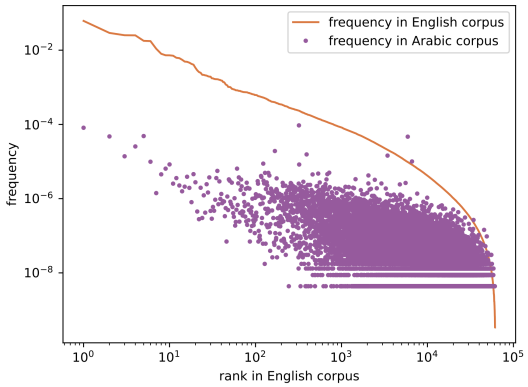
The results of this section suggest that code-switching, despite being infrequent, amounts to a non-negligible number of code-switched tokens in a large corpus that covers a large part of the most frequent words from the code-switched language, which might be sufficient to learn a mapping between the respective embeddings of two languages.

5.2 Results of `CoSwitchMap`

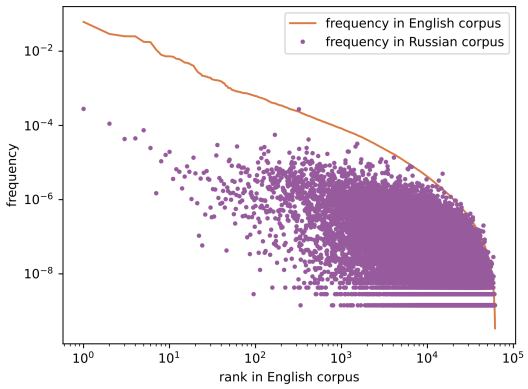
`CoSwitchMap` introduces a new way to learn a seed bilingual dictionary from code-switching. This seed dictionary can then be used as initialization for a self-learning loop. `CoSwitchMap` reuses the self-learning algorithm of VecMap. We thus compare the method to VecMap and other unsupervised mapping-based methods.

Wasserstein-Procrustes (**WP**) (Grave et al., 2018) is a method relying on optimal transport. The initial dictionary is provided through the convex relaxation of a graph-matching problem between the graphs, for each monolingual embedding, of similarities between each word. Self-learning is then performed. At each step, a new mapping is learned from a given dictionary with Procrustes as in most other methods. A new dictionary is obtained from a given mapping by solving an optimal transport problem using Wasserstein distance.

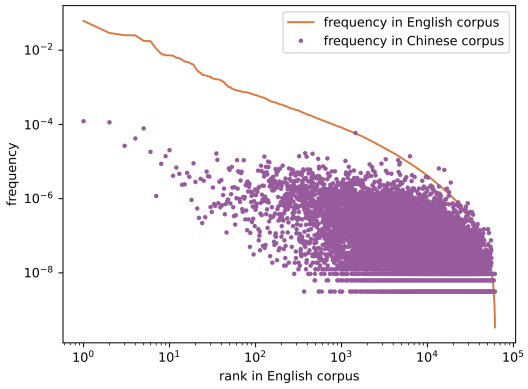
MUSE (Conneau et al., 2017) relies on adversarial learning. A linear mapping is trained to maximize the loss of a discriminator that is simultaneously trained to distinguish embeddings from both languages that are being aligned. The mapping is orthogonalized at each step using the same update as ours (cf. Equation 5). The obtained mapping is then refined with self-learning. Each new mapping is obtained with Procrustes. Each new dictionary is obtained through a nearest-neighbor search.



(a) Arabic



(b) Russian



(c) Chinese

Figure 2: Frequency of words from an English dictionary in the English corpus (line) and non-English one (dots) according to the rank in frequency in the English corpus.

VecMap (Artetxe et al., 2018b) relies, like WP, on graph-matching for initialization: each word is represented by a vector containing the distance to all other words. After taking the square root of each embedding matrix, sorting the values in each vector, and normalizing them, a nearest-neighbor search provides the initial dictionary. Self-learning then consists of Procrustes for learning each new mapping and nearest-neighbor search for learning each new dictionary.

method	en-ar	en-ru	en-zh
Methods with other self-learning procedures			
WP	$10.7_{\pm 9.9}$	$36.9_{\pm 1.4}$	$0.6_{\pm 0.8}$
MUSE	$30.9_{\pm 3.3}$	$41.7_{\pm 2.9}$	$0.0_{\pm 3.3}$
Different initializations for the same self-learning			
VecMap	$36.4_{\pm 1.8}$	**49.1**$_{\pm 0.4}$	$0.0_{\pm 0.0}$
w/ MUSE init.	$37.4_{\pm 2.6}$	$48.3_{\pm 0.4}$	$0.0_{\pm 0.1}$
w/ WP init.	$38.6_{\pm 0.7}$	$45.8_{\pm 2.8}$	$0.1_{\pm 0.0}$
w/ identical init.	$\underline{39.8}_{\pm 0.3}$	$\underline{48.9}_{\pm 0.2}$	$36.8_{\pm 0.8}$
CoSwitchMap (ours)	**39.9**$_{\pm 0.1}$	$\underline{49.0}_{\pm 0.3}$	**37.9**$_{\pm 0.9}$
supervised	*43.0*	*52.7*	*43.3*

Table 4: Comparison of CoSwitchMap with other unsupervised mapping-based methods. The score is the top-1 accuracy of a nearest-neighbor search with CSLS criterion for BLI. Results are averaged over 5 seeds and the standard deviation is provided (except for the deterministic supervised baseline). Bold indicates the best score for a given language pair and all scores that are within the standard deviation of the best one are underlined.

Søgaard et al. (2018b) showed that fully unsupervised mapping-based methods can fail in certain conditions, namely when languages are distant. They obtain better results using a seed dictionary built with identical words found in both vocabularies instead of one resulting from graph-matching algorithms or adversarial mapping that might rely too heavily on the need for isometry between embeddings. We use this initialization with VecMap self-learning to compare with ours and VecMap.

Table 4 shows how CoSwitchMap fares compared to the other aforementioned mapping-based methods in a Bilingual Lexicon Induction (BLI) task. For the three language pairs tested, fully unsupervised mapping-based methods (WP, MUSE, and VecMap) are outperformed or matched by CoSwitchMap. The gap is the most significant for the English-Chinese pair, where fully unsupervised methods largely fail, while initialization with identical words scores slightly behind CoSwitchMap. For the two other language pairs, the differences are less pronounced but CoSwitchMap is still among the best-performing ones.

In CoSwitchMap code-switching is used only for the initialization, the self-learning being the same as VecMap. Thus, Table 4 also compares different initializations with the same self-learning from VecMap. It must be noted that the initial-

method	results for different seeds				
WP	14.9	5.7	28.0	5.1	0.0
MUSE	34.1	33.9	26.5	32.4	27.3
Vecmap	37.8	37.4	35.9	33.2	37.9
id. init.	40.3	39.7	39.9	39.4	39.8
ours	40.1	39.8	39.7	39.5	40.1

Table 5: Breakdown of the BLI accuracy for each of the tested random seeds for the English-Arabic language pair. Each column represents a different random seed used for the algorithms.

ization methods of MUSE and WP provide better results when used with the VecMap self-learning method than with their original self-learning procedure. This validates the choice of the self-learning procedure for our method. But most importantly, it shows that the initialization provided by the code-switching training signal is significantly better than any other except the identical initialization for Arabic and Russian, and the original initialization for Russian. But CoSwitchMap always at least matches, if not outperforms, the best unsupervised baseline.

However, two things must be noted about the identical initialization. First, it might indirectly rely on code-switching, since the most frequently code-switched words will be present in the vocabulary of both languages[5]. Second, CoSwitchMap still outperforms this baseline for the English-Chinese pair, suggesting that explicitly relying on code-switching can sometimes provide more accurate alignment.

Table 4 also shows the results of a competitive supervised baseline, from the same framework as VecMap (Artetxe et al., 2018a) trained on a bilingual dictionary of 5,000 different words with their translations, distinct from those used for evaluation, but from the same origin (Conneau et al., 2017). Being unsupervised, CoSwitchMap is unsurprisingly outperformed by the supervised baseline, but falls short only by a few points, from 3.1 to 5.6. The supervised method has the unfair advantage of relying on a training bilingual dictionary, which is similar to the test dictionary used for evaluating BLI.

It is also worth noting that CoSwitchMap, along with all methods using VecMap self-learning, has results with a smaller standard deviation than the others. This suggests that there is a need for robust self-learning algorithms in unsupervised mapping-based methods. Table 5 shows the same algorithm can sometimes give different results according to the random seed used. WP and MUSE show more instability than methods with VecMap self-learning. However, it must be noted that the initialization might also play an important role in the stability of the results since VecMap provides slightly less stable results with its original initialization than with the two others (id. init. and ours).

[5]Only the most frequently code-switched words because vocabularies are usually truncated before alignment typically to 200,000 words

6 Conclusion

In a corpus like Wikipedia, code-switching is an infrequent signal that nonetheless involves a large portion of the most frequent vocabulary. It can thus be harnessed to learn cross-lingual word representations. We proposed CoSwitchMap to extract code-switching situations in an unsupervised manner and to use them to build a seed dictionary for learning a bilingual word embedding.

The method is limited to pairs of languages written in different scripts. But it is often for those pairs of languages that existing unsupervised methods fail, due to the languages being too distant. Our analysis shows that code-switched words seem to never have a frequency above a certain threshold, which suggests that a frequency-based method for code-switching extraction could be devised to adapt our method to pairs of same-script languages.

CoSwitchMap outperforms other unsupervised mapping-based methods in Bilingual Lexicon Induction for languages of different scripts. It shows that, with the right initialization, unsupervised mapping-based methods can work with distant languages. But, most of all, it demonstrates that code-switching can be valuable cross-lingual training signal.

7 Limitations

The reader should note that CoSwitchMap is thought of as a way to demonstrate the utility of code-switching as a cross-lingual signal, rather than as a method with direct practical utility. Indeed, the method only works for different scripts. It requires one to know the character ranges of the script at hand, which can still be seen as a very weak level of supervision, and which has prevented us from testing the method on a larger set of languages.

References

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018a. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019.

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

Terra Blevins and Luke Zettlemoyer. 2022. Language contamination helps explain the cross-lingual capabilities of english pretrained models.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomás Mikolov. 2016. Enriching word vectors with subword information. *CoRR*, abs/1607.04606.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *CoRR*, abs/1710.04087.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Stephan Gouws and Anders Søgaard. 2015. Simple task-specific bilingual word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1386–1390, Denver, Colorado. Association for Computational Linguistics.

Edouard Grave, Armand Joulin, and Quentin Berthet. 2018. Unsupervised alignment of embeddings with wasserstein procrustes. *arXiv preprint arXiv:1805.11222*.

Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Hervé Jégou, and Edouard Grave. 2018. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2984, Brussels, Belgium. Association for Computational Linguistics.

Jitin Krishnan, Antonios Anastasopoulos, Hemant Purohit, and Huzefa Rangwala. 2021. Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling. *CoRR*, abs/2103.07792.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Tomás Mikolov, Quoc V. Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA. Curran Associates Inc.

Libo Qin, Minheng Ni, Yue Zhang, and Wanxiang Che. 2020. Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. *CoRR*, abs/2006.06402.

Sebastian Ruder. 2017. A survey of cross-lingual embedding models. *CoRR*, abs/1706.04902.

Anders Søgaard, Sebastian Ruder, and Ivan Vulic. 2018a. On the limitations of unsupervised bilingual dictionary induction. *CoRR*, abs/1805.03620.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Shijie Wu and Mark Dredze. 2020. Do explicit alignments robustly improve multilingual encoders? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4471–4482, Online. Association for Computational Linguistics.

Min Xiao and Yuhong Guo. 2014. Distributed word representation learning for cross-lingual dependency parsing. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 119–129, Ann Arbor, Michigan. Association for Computational Linguistics.

Jian Yang, Shuming Ma, Dongdong Zhang, Shuangzhi Wu, Zhoujun Li, and Ming Zhou. 2020. Alternating language modeling for cross-lingual pre-training. In *AAAI Conference on Artificial Intelligence*.

A Character ranges for each language

For each of the languages experimented, we use the following character ranges:

- English: [a–zA–Z]
- Arabic: [\u0621–\u064A]
- Russian: [\u0401\u0451\u0410–\u044f]
- Chinese: [\u4e00–\u9fff\u3400–\u4dbf \U00020000–\U0002a6df\U0002a700–\ U0002ebef\U00030000–\U000323af\ufa0e \ufa0f\ufa11\ufa13\ufa14\ufa1f\ufa21\ufa23 \ufa24\ufa27\ufa28\ufa29\u3006\u3007][\ ufe00–\ufe0f\U000e0100–\U000e01ef]?

The regular expressions in English, Arabic and Russian are simple ranges. For Chinese (Han script), a more complex regular expression is used as it must take into account variation selectors that can be added after an ideogram. The whole regular expression was obtained from https://ayaka.shn.hk/hanregex