Proceedings of the LREC 2020 – 4th Workshop on Computational Approaches to Code Switching, pages 26–35
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020
European Language Resources Association (ELRA), licensed under CC-BY-NC

Evaluating Word Embeddings for Indonesian–English Code-Mixed Text Based on Synthetic Data

Arra'di Nur Rizal and Sara Stymne
Department of Linguistics and Philology
Uppsala University
Arradinur.Rizal.4511@student.uu.se, sara.stymne@lingfil.uu.se

Abstract

Code-mixed texts are abundant, especially in social media, and poses a problem for NLP tools, which are typically trained on monolingual corpora. In this paper, we explore and evaluate different types of word embeddings for Indonesian–English code-mixed text. We propose the use of code-mixed embeddings, i.e. embeddings trained on code-mixed text. Because large corpora of code-mixed text are required to train embeddings, we describe a method for synthesizing a code-mixed corpus, grounded in literature and a survey. Using sentiment analysis as a case study, we show that code-mixed embeddings trained on synthesized data are at least as good as cross-lingual embeddings and better than monolingual embeddings.

Keywords: Code-mixed data, word embeddings, synthetic corpora, code-mixed embeddings

1. Introduction

People from around the world are able to connect and exchange information instantly, through the internet and social media. This exchange of information is dominated by the English language (Danet and Herring, 2003; Kramarae, 1999; Poppi, 2014). To prepare Indonesians to go global, the English language has been taught to Indonesian students from elementary school. This exposure to the English language instigates frequent Indonesian–English code-mixing in Indonesia (Brown, 2000, p. 139). This phenomenon is clearly observable in social media not only used by Indonesians but also across languages (Cárdenas-Claros and Isharyanti, 2009; Shafie and Nayan, 2013). Code-mixed text is a challenge for the computational linguistic community, where work based on social media text (Chakma and Das, 2016; Barman et al., 2014) is common. This poses a challenge because most models such as word-embedding models assume the training data is monolingual.

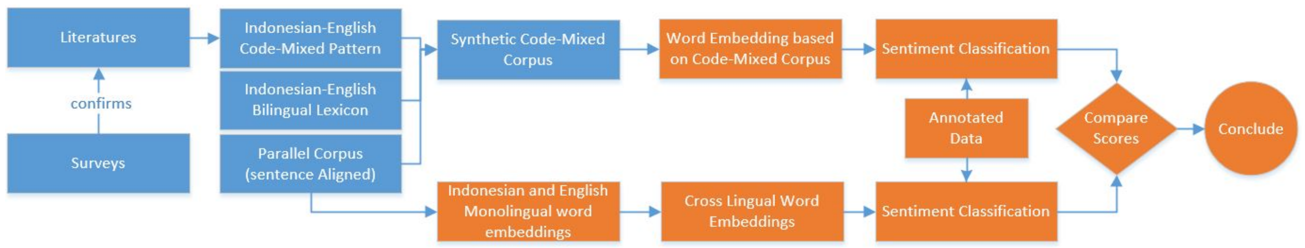
In this paper, we focus on code-mixed Indonesian–English text. Code-mixing has also been referred to as intra-sentential code-switching (Hoffmann, 2014), i.e. the two or more languages are mixed within sentences, not only between sentences. In code-mixed sentences, depending on the language, the word in L1 (e.g. Indonesian) is usually not only replaced with its translation in L2 (e.g. English), but can also be merged with affixes of the L1 (e.g. Indonesian). For instance "*Kita perlu* **revise document***nya*" (*ID: Kita perlu memperbaiki dokumennya*; **EN: We need to revise the document**). In the example, the English word **"revise"** is used instead of the Indonesian word **"memperbaiki"** and the English word **"document"** is merged with the Indonesian suffix **"-nya"**.

There is plenty of research on cross-lingual word-embeddings, which can use either monolingual corpora or parallel corpora to do projection, mapping or alignment (Ruder et al., 2017). In most cases, a set of monolingual embeddings in one language is projected to a set of monolingual embeddings in the other language, or both sets are projected into a shared space. These methods might not be enough to capture intra-sentential code-switched words since cross-lingual embedding tries to merge word representations from two sets of monolingual texts. Mixed words will therefore not be represented in cross-lingual word-embeddings. To address this issue, we suggest that a cross-lingual word embedding model based on code-mixed sentences might be needed. We will call these embeddings code-mixed word embeddings. These word embeddings still cover more than one language like cross-lingual embeddings, but they do so in a setting where the languages are mixed, rather than separate. This has previously been proposed for English–Spanish by Pratapa et al. (2018b).

To train a word embedding model of any type, a large amount of data is needed. Crawling social media does not guarantee that we get balanced corpora of diverse patterns of code-mixed sentences, nor that we get a large enough set of code-mixed sentences. To avoid getting a skewed corpus, we need to be able to control class (code-mixed pattern) distribution in our corpus. One possible method is to synthesize the training corpus By synthesizing a corpus, we will be able to control the class distribution in our data set, and we can easily create a large corpus.

A number of studies has previously proposed methods for synthesizing code-mixed text, using a variety of approaches, based on neural networks (Winata et al., 2019; Chang et al., 2019), linguistic theories (Lee et al., 2019; Pratapa et al., 2018a), or heuristics (Wick et al., 2016). None of these studies have incorporated mixed morphology, which is important in the Indonesian–English setting we are interested in. Our study is using a heuristic approach similar to the work by Wick et al. (2016), which is, however, focused on artificial code-switching involving several languages with the end goal of improving cross-lingual NLP, rather then on mimicking naturally occurring code-mixed data with the end goal of processing code-mixed data.

The main purpose of this study is to evaluate whether code-mixed sentences can be better represented by code-mixed embeddings than by cross-lingual embeddings based on



Literatures
confirms
Surveys
Indonesian-English Code-Mixed Pattern
Indonesian-English Bilingual Lexicon
Parallel Corpus (sentence Aligned)
Synthetic Code-Mixed Corpus
Word Embedding based on Code-Mixed Corpus
Sentiment Classification
Annotated Data
Compare Scores
Conclude
Indonesian and English Monolingual word embeddings
Cross Lingual Word Embeddings
Sentiment Classification

Figure 1: Overview of the methodology for this study. Blue color shows the process of synthesizing a code-mixed corpus. The orange color shows the process of Word Embedding Evaluation

monolingual embeddings. In addition, as a prerequisite task, a method for synthesizing Indonesian-English code-mixed corpus will be presented, grounded in literature and a survey, but simpler than previously proposed methods. The main contributions of this paper are:

- A simple method for creating a synthetic code-mixed corpus of high quality
- Evaluation of code-mixed, cross-lingual and monolingual word embeddings on code-mixed Indonesian-English text on a sentiment classification task.

The methodology that we used is shown in Figure 1. The process is divided into two phases. The first one is to synthesize a code-mixed corpus. The second phase is word embedding evaluation, where we compare different types of word embeddings on a sentiment classification task, in order to investigate the feasibility of using word embeddings created from a synthetic code-mixed corpus.
The paper is organized as follows: In section 2 we review related work. In section 3 we describe Indonesian–English code-mixing and section 4 describes the data used. Section 5 describes the synthesis of the code-mixed corpus, including the results on a survey of code-mixed patterns and an evaluation. Section 6 describes how we trained word embeddings, and section 7 gives the results for these word embeddings on a sentiment classification task. The paper ends with a conclusion and suggestions for future work in section 8.

2. Related Work

We are only aware of two attempts to compare monolingual word embeddings, code-mixed word embeddings, and cross-lingual word embeddings trained on corpora with different types of contents. Pratapa et al. (2018b) tested different word embedding techniques on a code-mixed corpus, namely correlation based model, compositional model, and skip-gram model on Spanish–English. They created the bilingual word embedding using monolingual embedding and synthetic code-mixed corpus. They evaluated these word embeddings on a semantic task (sentiment analysis) and a syntactic task (POS tagging). They found that word embeddings created from the code-mixed text, even if it is artificially created, is needed for processing code-mixed text since the existing cross-lingual embeddings are not suitable. Our study is applied to a different language pair, and we use different methods for training embeddings. We were unable to evaluate on a syntactic task, due to a lack of annotated data for code-mixed Indonesian–English.
Wick et al. (2016) train code-mixed embeddings on synthetic data where five languages are mixed. They evaluate these embeddings on a bilingual analogy tasks and on cross-lingual sentiment analysis. The goal of this work is different from us, since their purpose was to find embeddings that are useful in a cross-lingual learning setting, rather than in a setting where code-mixed data is processed.
As for non-synthetically creation of code-mixed corpus, there have been multiple attempts such as Turkish–German Code-Switching Corpus (Çetinoğlu, 2016), Arabic–Moroccan Darija Code-Switched Corpus (Samih and Maier, 2016), Hindi–English Code-Mixed Corpus (Vijay et al., 2018), and English–Spanish Code-Switching Twitter Corpus (Vilares et al., 2016). Yet, English–Indonesian Code-Mixed Corpus does not exist. Most of the code-mixed corpora are created by fetching social media information such as Tweets.
There are several studies that precede our study in code-mixed text synthesis. Some of the recent approach use neural networks. Winata et al. (2019) create a sequence-to-sequence model with a copy mechanism which learn how to combine sentences from parallel corpora to generate code-mixed text. Chang et al. (2019) utilize a generative adversarial network to generate code-mixed sentences.
One early study which synthesizes code-mixed sentences is the work by Wick et al. (2016). They present a method for creating artificially code-switched text across many languages, which they apply to five languages. They use an algorithm where they randomly sample words to be replaced. For each sampled word, they pick one of the concepts, or word senses, possible for that word, and sample a word in any language that belongs to that concept. Our method only mixes two languages, and it is even simpler, in that it does not require a concept dictionary. We also integrate morphology mixing, which is not handled by Wick et al. (2016). Furthermore, the purpose of the two studies are different, since our goal is to mimic naturally occurring code-mixing, whereas their goal is to create data which is useful for cross-lingual learning.
Lately, theories about the nature of code-switched discourse have been used to generate synthetic code-mixed text. Although there is no consensus, there are, among others, three leading theories explaining the formation of code-

switched text. They are the Functional Head Constraint (FHC) theory (Di Sciullo et al., 1986; Belazi et al., 1994), the Equivalence Constraint (EC) theory (Sankoff, 1998; Poplack, 1980) and the Matrix Language Frame (MLF) theory (Myers-Scotton, 1997).
One study which uses EC theory to synthesize the code-mixed test is the work by Pratapa et al. (2018a). The basic idea of EC-based code-mixed sentence generation is to ensure that the generated sentence does not break monolingual grammar in both languages. For instance, EC theory will disallow the fragment *book blue* in an English sentence since it is grammatically incorrect. They apply EC-theory to generate synthetic sentences by collapsing two constituency parses tree into one. In order to apply their method, they create a parallel word-aligned corpus that is used to replace every L2 words with L1 words in L2 constituency tree with a corresponding hierarchical structure, on which they can apply the EC-theory to maintain the grammatical structure of L2. In contrast, our study uses a simpler rule-based method replicating the pattern that human produces. It does not need parse trees nor word aligned corpora, but a bilingual dictionary.
Lee et al. (2019) present a study where they use the MLF theory to generate synthetic code-mixed text. MLF theory suggests that in a code-mixed sentence, there will be a dominant language (matrix language) and inserted language (embedded language). This study uses parallel text data aligned at the phrase-level, as a basis for inserting phrases from the embedded language into the matrix language. They apply their method to language modelling, where they sample phrases to insert on the fly, and also combine this with naturally occurring code-mixed data. The idea of using embedded language to create synthetic code-mixed text is similar to our study. An important difference is that their method is based on parallel data whereas our method is based on a bilingual dictionary. They do not model morphology mixing.
There is some work on sentiment classification for code-mixed data. Typical methods are text normalization (Sharma et al., 2015) or adding additional annotations (Vilares et al., 2016).

3. Code Mixing in Indonesian

The Indonesian language has two distinct forms, a formal and an informal variant. Code-mixing can occur with both variants, both in speech and in written format, e.g. magazine articles, text messages, and social media content. Most social media text, such as twitter, use the informal variant.
The informal form of Indonesian is mainly categorized into two groups. The first group is the informal usage of words. For instance; the usage of informal pronouns (e.g. *"gue"* instead of *"saya"*; **EN: "I"**), informal abbreviations (e.g. *"lht"* instead of *"lihat"*; **EN: "see"**), and non-standard spelling (e.g. *"haaaaloo"* instead of *"halo"*; **EN: "hello"**). The second group is the informal grammar especially the informal use of affixes. In the informal form, formal affixes are either dropped or replaced. For instance; *"saya menjual ayam"* (**EN: "I am selling chicken"**) becomes *"saya jual ayam"* (the prefix *men-* is dropped) or *"saya ngejual ayam"* (the prefix *"men-"* is replaced with informal prefix *"nge-"*). Another example is *"kirimkan paket ini"* (**EN: "send this package"**) becomes *"kirim pake ini"* (suffix *-kan* is dropped) or "kirim*in* paket ini" (suffix *-kan* replaced with informal suffix *-in*). In code-mixed sentences, both categories of informality can be used. For instance; "*Gue nge***update document**"; *ID: "saya memperbaharui dokumen"*; **EN: "I am updating a document"**. In the example, the formal pronouns *"saya"* is replaced with the informal pronouns *"Gue"* and the informal prefix *"nge-"* is used with english word **"update"** to replace the formal verb *"memperbaharui"*.
Many studies have been conducted to analyze the usage and the form of English–Indonesian code-mixed sentences (Marzona, 2017; Siregar et al., 2014; Kurniawan, 2016; Habib, 2014; Setiawan, 2016). From these studies, we deduced that there are two main forms of Indonesian code-mixed sentences. These two patterns appear in all literature. The first form is word replacement where, for instance, Indonesian nouns, adjectives, verbs, and conjunctions are replaced with their English counterparts. The second form is morphology mixing where Indonesian affixes (formal and informal) are mixed with English verbs. For instance, "*Dokumennya bisa di***download anytime**" (*ID: Dokumennya bisa diunduh kapan saja*; **EN: The document can be downloaded anytime**). The study by Kurniawan (2016) found that 60 percent of English words in an English–Indonesian code-mixed sentence is the average distribution in code-mixed Indonesian–English text. However, this is based on a small study of only three young persons.
In this work, we will only be concerned with the formal variant, since the formal variant has smaller word variation than the informal variant. Formalizing the corpus helps reduce the Out-of-Vocabulary (OOV) when we do a downstream task such as sentiment analysis. Furthermore, the sentiment analysis corpus (Saputri et al., 2018) we use mostly contains formal Indonesian. However, all methods used would be equally applicable to informal Indonesian.

4. Data and Pre-Processing

We use multiple English–Indonesian sentence-aligned corpora from several genres: The Open Subtitle 2018 Corpus (Lison et al., 2018), TED 2018 Corpus (Cettolo et al., 2012), and GlobalVoice Corpus (Tiedemann, 2012), all taken from the OPUS collection (Tiedemann, 2012). These corpora are cleaned and pre-processed. Table 1 gives an overview of the size of the corpus before and after pre-processing. Open Subtitle 2018 dominates the corpus.
The open subtitle corpus contains a lot of non-letter characters (e.g. ¶\∗#) and formatting (e.g. {\cHFFFFFF}). To clean this, we adjusted the PrepCorpus script[1] to accommodate the Indonesian translation. Moreover, the subtitles also contain song lyrics that are not translated into Indonesian. Therefore, the non-translated sentence pairs were removed from the corpus. These sentences were automatically removed leveraging a bilingual lexicon. A sentence pair is removed if more than 60 percent of the words in the Indonesian sentence are found in the English dictionary. The

[1] `https://github.com/rbawden/PrepCorpus-OpenSubs`.

Original	TED 2018	Glove	OpenSub	Total
Sentences	117,359	14,448	9,268,181	9,399,988
Tokens – ID	1,646,944	238,968	47,025,227	48,911,139
Tokens – EN	1,882,869	264 689	54,969,761	57,117,319
Cleaned	TED 2018	Glove	OpenSub	Total
Sentences	114,915	14,213	9,237,234	9,366,362
Tokens –ID	1,624,189	232,060	45,943,213	47,799,462
Tokens – EN	1,862,658	257,976	53,723,799	55,844,433

Table 1: The size of corpora before and after pre-processing, for Indonesian (ID) and English (EN).

TED2018 corpus has meta-data information in the corpus. For example <speaker>Al Gore</speaker>. We removed all such meta-data as well as blank lines and double dash characters (--). The Global Voice Corpus contains non-letter characters (e.g. # and ...). These characters as well as blank lines were also removed. After this basic cleaning, we extended all English contractions, (e.g I've → I have) using our script which based on the pycontractions library[2].

Several Indonesian sentences in the corpus were formed in an informal manner. Since this study only concerns formal Indonesian, formalization of the informal sentences was done during the pre-processing. To formalize the informal sentence, we are using the Colloquial Indonesian Lexicon[3] as well as a lexical normalization method from the study by Barik et al. (2019). The lexicon and the normalization method are sufficient for mapping the informal words and affixes back to the standard form. We then lower-cased all data. For English we used Moses (Koehn et al., 2007) to tokenize the data, and for Indonesian, the InaNLP toolkit (Purwarianti et al., 2016). To make sure that the character replacement in Indonesian is consistent with the English Corpus (e.g. apostrophe (') → &apos;), the tokenized corpus was re-tokenized using Moses with the English language as an option. The sentences were not lemmatized because affixes are part of the Indonesian code-mixed pattern.

For synthesizing our code-mixed corpus, and for some of the methods for creating cross-lingual embeddings, as well as for the cleaning described above, we used the bilingual Indonesian–English lexicon from Facebook's ground truth bilingual dictionaries[4]. The bilingual lexicon contains 96,518 words pairs.

5. Synthesizing a Code-Mixed Corpus

In this section, we describe the work on synthesizing the code-mixed corpus. We first describe a survey conducted to investigate the patterns of code-mixing actually in use across Indonesia. We then describe our method for synthesizing and evaluate the resulting corpus.

5.1. Survey

The purpose of the survey is to confirm the code-mixed patterns described in the literature. The goal is to validate whether Indonesian people from different cities would create English–Indonesian code-mixed sentences in the same way. The reason for this is that the literature are typically based on one specific subset of the Indonesian population, e.g. one city (Siregar et al., 2014), one institution (Kurniawan, 2016) or one industry (Marzona, 2017). Therefore, since Indonesian society is not homogeneous, these studies might not represent more than a single variant of Indonesian.

The survey was given out to Polyglot Indonesia members. Polyglot Indonesia is a non-profit organization in Indonesia, which started as a community for language enthusiasts. The organization's members reside across Indonesia.

The survey was made in Google Form which was then spread via Polyglot Indonesia's WhatsApp group, which has 200 members from various cities across Indonesia. Polyglot Indonesia members were chosen because they have linguistic knowledge and they have a diverse cultural and linguistic background. The survey has two questions. 1. The city of origin; 2. Write 10 examples of code-mixed sentences. The survey was conducted for 4 weeks starting in September 2019.

The respondents of the survey (115 people) reside in 14 cities around Indonesia. All of them produced code-mixed sentences with the two patterns that appear in all literature (Marzona, 2017; Siregar et al., 2014; Kurniawan, 2016; Habib, 2014; Setiawan, 2016). For instance "*Harganya ga* **reasonable**" (*ID: Harganya tidak masuk akal*; **EN: The price is not reasonable**) and "*Gue gampang ke***distract** *gitu*" (*ID: saya gampang terganggu*; **EN: I am easily got distracted**). In the two examples, the underlined word is the informal form.

The results of the survey confirm that the code-mixed patterns found in the literature actually reflects the code-mixed patterns used across Indonesia well. The survey also confirms, referring to Matrix Language Frame theory, Indonesian language is the dominant language in Indonesia-English code-mixed text. We thus go on to develop a method implementing the two main patterns: to exchange words and to exchange words with the addition of Indonesian affixes.

5.2. Method for synthesis

We developed an algorithm for the code-mixed sentence synthesizer, based on a monolingual Indonesian corpus and

[2]`https://github.com/ian-beaver/pycontractions`.

[3]`https://github.com/nasalsabila/kamus-alay`.

[4]`https://dl.fbaipublicfiles.com/arrival/dictionaries/id-en.txt`.

Algorithm 1 Code-mixed synthesis algorithm

```
for each Indonesian sentences do
    swapped ← 0
    for each word in sentence do
        if swapped / num_words_in_sentence < MAX_SWAP then
            continue with next sentence
        end if
        if random.generate() > SWAP_THRESHOLD then
            strippedWord, affixes ← removeAffixes(word)
            if word in bilingual dictionary then
                swap(word, englishWord)
                swapped++
            else if strippedWord in bilingual dictionary then
                mergedWord ← addAffixes(englishWord, affixes)
                swap(word, mergedWord)
                swapped++
            end if
        end if
    end for
end for
```

a bilingual dictionary. The reason for basing the synthesis on the Indonesian side is that code-mixed Indonesian–English tend to follow Indonesian syntax. Only formal Indonesian was addressed, but the algorithm can easily be extended to cover informal Indonesian as well.

Algorithm 1 shows how we synthesize the code-mixed corpus. For each sentence, we go through the words in the sentence from left-to-right. For each word, we try to exchange it with a probability set by SWAP_THRESHOLD. If the Indonesian word is found in the bilingual lexicon, it is exchanged (swapped) with the English word, otherwise, we try to strip it of its affixes, and lookup the stem. If the stem is found, we merge the affixes to the English stem, and exchange this mixed word with the original word. Otherwise, no exchange takes place. We limit the number of words exchanged in any sentence to MAX_SWAP. The list of affixes is based on the Indonesian affixes description from the study by Adriani et al. (2007) (section 2). It contains 30 affixes of the following types:

- Inflectional suffixes, for example, ”-kah”, ”-lah”, ”-tah”, ”-pun”, ”-ku”, ”-mu”, and ”-nya”,
- Derivational prefixes, for example, ”be-”, ”di-”, ”ke-”, ”me-”, ”pe-”, ”se-”, and ”te-”,
- Derivational suffixes, for example, ”-i”, ”-kan”, ”-an”
- Derivational confixes, for example, ”be-an”, ”me-i”, ”me-kan”, ”di-i”, ”ke-an”.

In our experiments we set SWAP_THRESHOLD to 50% and MAX_SWAP to 60% which is the average percentage of English words used in a code-mixed sentence in the study of Kurniawan (2016). It is also close to the average value in the code-mixed sentences in the corpus taken from the survey result, which is 55.1%. Other values for these two parameters are possible, but we leave the investigation of this effect to future work. The generated code-mixed corpus has the same size as our parallel corpus, over 9M sentences and 47M words. Some examples are shown in Table 2.

5.3. Corpus Evaluation

To measure the similarity between the real code-mixed sentences and the synthetic code-mixed text, we use two previously proposed measurements: Switch-Point Fraction (SPF) and Code Mixing Index (CMI). SPF (Pratapa et al., 2018a) measures the number of switch-points in a sentence divided by the total number of word boundaries . We define ”switch-point” as a point in a sentence at which the words switch to another language. CMI (Gambäck and Das, 2014) measures the number of switches at the utterance level. It is computed by determining the dominant language (the most frequent language) and then counting the frequency of words belonging to the embedded language. We calculate SPF and CMI at the corpus level, averaging the SPF and CMI for all sentences in a corpus.

We used these two measures to investigate if the synthesized corpus seemed to have the same characteristics as a naturally occurring code-mixed corpus. As a point of comparison, we used the corpus produced from the survey. Table 3 shows that the natural and synthesized corpora have very similar values for both measurements, indicating that the distribution of words in the code-mixed corpus mimics that of naturally occurring code-mixed sentences. It also indicates that the choices for the tunable values of the synthesizing algorithm were well chosen.

To further make sure that the generated sentences are good examples of code-mixed text, we conducted a human evaluation. Two native Indonesian bilinguals (Indonesian and English) evaluated 500 generated sentences, chosen at random. 86 percent of the evaluated sentences were considered to be acceptable code-mixed sentences while 14 percent of the sentences were considered to be incorrect, e.g phrases were not translated properly, or not properly formed (e.g. misspelling), or words had translations that are incorrect in that context. For example, the phrase

Good Conversion.	
1	ID: Dia paham betul kisah Irak, mungkin melebihi siapapun. EN: He knows the story of Iraq perhaps more than anybody else CM: Dia paham betul story iraq, probably melebihi anyone .
2	ID: Anda harus meninggalkan misi ini. EN: You have to leave this mission. CM: Anda harus leaving misi ini.
3	ID: Foto oleh Forum Pengada Layanan. EN: Photo by Forum Pengada Layanan. CM: Photo by Forum Pengada Layanan.
4	ID: Dua belas dari pemerkosanya diduga kini telah ditangkap, tapi dua lagi masih buron. EN: Twelve of the suspected rapists have now been arrested, but two are still at large. CM: Dua belas dari rapistnya diduga kini have captured, but two more still buron.
5	ID: Kesehatannya memburuk sejak kematian putrinya. EN: She is not doing so well since the death of her daughter. CM: Kesehatannya memburuk since deaths daughternya.
Bad Conversion.	
1	ID: Dia memusnahkan prestasi ekonomi. EN: He destroyed economic achievement. CM: Dia gutted prestasi economy.
2	ID: Adik saya menjawab Kamu tidak mengerti sama sekali. EN: And my sister replies You do not understand anything. CM: Adik saya answer kamu tidak understand equal once.
3	ID: Dia mengambil risiko hidup membujang seumur hidup. EN: She risks living the rest of her life alone. CM: Dia retrieving risks alive membujang seumur alive.

Table 2: Example sentences from the generated synthetic corpus. ID: Indonesian, En: English: CM: Synthesized code-mixed.

Measurement	Survey	Synthetic
SPF	0.2951	0.2989
CMI	0.8559	0.8431

Table 3: Switch-Point Fraction and Code Mixing Index from the corpus taken from survey result and the corpus from synthetic code-mixed text.

"*ID:Terima kasih*" sometime becomes "**EN:accept love**" or "**EN:accept** *ID:kasih*" where it is supposed to be replaced by "**EN:thank you**". To evaluate the inter-evaluator agreement, Cohen's kappa (Cohen, 1960) was used. The kappa coefficient is 0.874 suggesting that both evaluators are in almost perfect agreement (between 0.81 and 1.00) in the evaluation (Landis and Koch, 1977).

Table 2 shows examples of code-mixed sentences judged as good and bad. The main issue with the three bad examples is that word by word translation causes the sentences to be incomprehensible. In example 1, the word *"memusnahkan"* is replaced with the word *"gutted"*, which is not contextually correct. This is because the synthesizer does not understand the context of the sentence nor understand word sense. In example 2, the phrase *"sama sekali"* (**EN: "anything"**) is replaced by the fragment **"equal once"**. This example shows another example of a phrase that is not translated properly. In example 3, the fragment *"hidup membujang seumur hidup"* which is supposed to be replaced with the fragment **"living the rest of her life alone"** is replaced with the fragment "**alive** *membujang seumur* **alive**". In addition, the word *"mengambil"* is supposed to be deleted or not replaced. The conversion for example number three is considered more complex since the synthesizer needs to be able to understand the sentence context and word alignment.

Overall, the majority of issues come from the inability of the synthesizer to replace phrases, which sometimes render sentences unintelligible. Another common issue is the inability to add function words when it is needed. For example, "*ID: ingin tahu*" became "**EN:want know**" where it should became "**EN:want to know**". Misspelling is also a common issue when replacing words and adding affixes. A misspelled word leads to the creation of multiple variants of the same word in the word embedding set, which is not only useless but also reduces the potential embedding quality of the correct word. If the misspelled word were correct, there would be more training data. We think that this issue could be solved with a spelling correction algorithm.

The code-mixed synthesis algorithm is simple, but it can still capture the two types of code-mixed patterns we have identified. As shown, it does overall give good results. There are plenty of options for improvement, though, such as tuning the two thresholds, choosing the words to exchange at random, rather than linearly, and basing the words to exchange on part-of-speech tags. Many of the errors are due to multi-word expressions and collocations, so identifying these, or at least applying the algorithm on the phrase-level could also potentially lead to improvements. Nevertheless, we do believe that the quality of the corpus is good enough for training code-mixed word embeddings.

6. Training Word Embeddings

We first train monolingual embeddings for each language based on each monolingual side of the parallel corpus using FastText[5] (Joulin et al., 2016) with default configuration. The model chosen to create word embeddings is the skip-gram model (Mikolov et al., 2013).[6] The code-mixed word embeddings were created in the same way as the monolingual embeddings, but from the synthetic code-mixed corpus using the same tool (fastText) and configuration as for the monolingual embeddings.[7]

To create cross-lingual word embeddings, we used Vecmap[8] (Artetxe et al., 2018) to combine the Indonesian and English monolingual embeddings into a shared space. We used four different modes provided by Vecmap: supervised, semi-supervised, identical, and unsupervised, with the default configuration. The supervised mode, the semi-supervised mode, and the identical mode are trying to learn how to map two monolingual embeddings to a shared space using seed dictionaries. The difference is on how each mode creates seed dictionaries. The supervised mode uses the full bilingual lexicon of 96,518 word pairs to create a seed dictionary. The semi-supervised mode is intended to be used if there is no large bilingual lexicon. The small number of word pairs will be used to bootstrap the training. In our case, we use the same bilingual lexicon as for the supervised mode to bootstrap the training. The identical mode uses identical words from both sets of monolingual word embeddings to build the seed dictionary. The unsupervised mode does not require a seed dictionary building. Instead, it uses unsupervised initialization based on the isometry assumption of monolingual embeddings. The isometry assumption assumes that the embedding spaces are perfectly isometric.

Since Vecmap is intended to be used for tasks like Machine Translation or zero-shot transfer for a range of tasks, where the texts they are applied to are monolingual, the output of Vecmap are two files (source language and target language). To apply these embeddings to code-mixed data, we need a single set of embeddings. To do this, we added all English words that did not occur in Indonesian to the Indonesian embeddings. This means that word forms that happen to occur in both languages will get the embedding from the Indonesian side. The reason that we prioritized Indonesian for shared words, is that the corpus is syntactically Indonesian. However, other strategies are possible, like averaging the embeddings, but we leave this for future work.

Compared to creating cross-lingual embeddings based on monolingual word embeddings, creating code-mixed word embeddings is straightforward and a lot faster.[9]

7. Evaluating Word Embeddings

As a final step, we evaluate the different word embeddings on a sentiment classification task. In this section, we describe the sentiment data, the architecture of our sentiment classifier, and the results of the evaluation.

7.1. Sentiment Classification Task

We use the sentiment classification tweets-emotion corpus (5,000 sentences) by Saputri et al. (2018)[10]. We use this corpus because it contains Indonesian–English code-mixed sentences.

For this study, the original code-mixed corpus which has 6 classes (anger, happiness, sadness, fear, love) were converted into two polarities (positive and negative). Anger, sadness, and fear were converted into negative, while happiness and love were converted into positive. The reason for converting the corpus into a binary sentiment task is that it is very small and the classes for the six-way classification had few training examples. The preparation of code-mixed corpus is similar to the Indonesian Corpus. Sentences were formalized, if needed, lower-cased, and tokenized.

7.2. Sentiment Classifier Architectures

Note that the purpose of this study is not to achieve state-of-the-art on sentiment classification, but to compare word embeddings. For that reason, we use a rather simple architecture for classification. In preliminary experiments, we also tried deeper architectures, but they tended to overfit since the data is quite small. To further highlight the strength of each embedding type, the weights of word embeddings were purposely frozen, i.e. not updated, during training.

The classifier architecture used is the Deep Averaging Network architecture by Iyyer et al. (2015) as shown in Figure 2. Each word in the input sentence was converted into its vector representation using word embeddings. Then, from this word representation, a sentence representation was created by averaging the representation of each word. The sentence representation was then fed to multiple fully-connected layers with ReLU activation followed by a softmax layer. A lambda layer was used to create sentence representation. The implementation was done using Keras framework.[11] Cross-entropy was used as a loss function with the Adam optimiser (lr=0.00003, β_1=0.9, β_2=0.999). The training process was done for 20 epochs (the model tends to overfit after 20 epochs) with 32 as batch size. Standard 10-fold cross-validation with a split of 90 percent for the training set and 10 for test set split was used. We used the F1-score as a metric.

7.3. Word Embedding Evaluation Result

Table 4 shows the scores using the different types of word embeddings. The score is the mean and maximum of model

[5]`https://github.com/facebookresearch/fastText`.

[6]According to fasttext documentation, skipgram model is better in performance compared to CBOW in practice.

[7]There was no need to use parallel data for these embeddings, the reason for this was that we wanted the two sets of monolingual embeddings to be comparable. The quality of all these sets of embeddings could be improved by also adding additional monolingual data.

[8]`https://github.com/artetxem/vecmap`.

[9]25 minutes compared to 6 hours on an 8 Intel Xeon processor (E5620) with 4 cores @ 2.40Ghz with 192GB RAM

[10]`https://github.com/meisaputri21/Indonesian-Twitter-Emotion-Dataset`.

[11]`https://github.com/keras-team/keras`

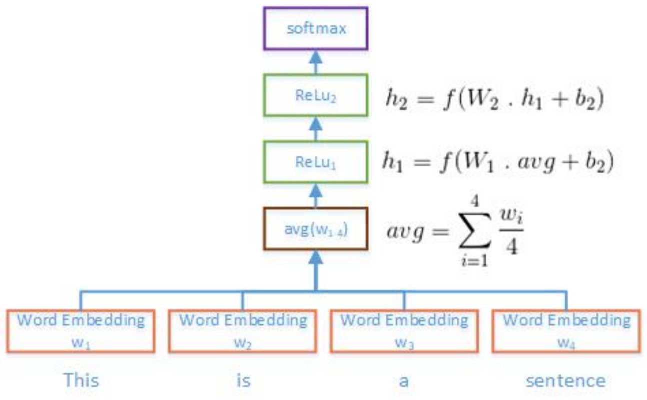


Figure 2: Sentiment Classification Architecture using Deep Averaging Network (Iyyer et al., 2015)

F1-score from each fold, accompanied by the standard deviation (STD). Code-mixed embeddings clearly give an advantage compared to using monolingual embeddings, with an average improvement of 0.55% over Indonesian and 0.82% over English. It even gives slightly better results than all of the cross-lingual embedding modes, which requires more processing times and a bilingual lexicon (supervised and semi-supervised). It is also worth noting that the differences between the different types of initialization for the cross-lingual embeddings are minor, showing no advantage of supervision over the unsupervised variants.

Analyzing the word embeddings, we find that the code-mixed word embeddings have a somewhat higher number of OOVs than the cross-lingual embeddings, see Table 5. This is due both to missing English and Indonesian words, which were on one side of the original corpus, but where the English word was never chosen by the swap operation, or the Indonesian word was always swapped. We do note that some words with mixed morphology are covered in the code-mixed embeddings, such as "**driver***nya*" (EN: *The driver*) and "**viral***kan*" (EN: *make it viral*). In this context, we note that the code-mixed embeddings are actually trained on less data since it is trained on the synthesized Indonesian corpus of 9.4M sentences, whereas the cross-lingual embeddings are trained on 9.4M sentences for each Indonesian and English (18.8M sentences in total), which contributes to this issue. This means that the competitive results on the sentiment analysis task are not due to better coverage of words. We think that the reason code-mixed embeddings give a slightly better score than cross-lingual methods, despite the slightly higher OOV rate and fewer types, is because it has an inherent representation of how both Indonesian and English words are related to each other within the code-mixed context. This feature seems not to be fully captured or replicated by cross-lingual methods.

8. Conclusion and Future work

In this study, we show that code-mixed embeddings are competitive and even slightly better than both monolingual and cross-lingual embeddings on a sentiment analysis task of code-mixed Indonesian–English. A large code-mixed corpus is needed to train these embeddings, but we show that such a corpus can be synthesized based only on a monolingual Indonesian corpus and a bilingual lexicon. We design a simple method for synthesis, which is, however, firmly grounded in both theory and a survey among speakers across Indonesia. The synthesis resulted in 86% acceptable sentences, which we show was enough for training competitive code-mixed embeddings. In this work, we only explored one method for training word embeddings. We think that it would be useful to explore other methods, as well, in order to see how well they work for code-switched data, and to evaluate the embeddings also on other tasks than sentiment analysis. The synthesis method could also be potentially improved in many ways, most importantly by extending the matching from single words to multi-word expressions. Other more resource-intensive options could be to integrate spell checking or a POS-tagger into the synthesis. We also want to further explore the impact of training set size for the different types of embeddings, which might have given an unfair advantage of cross-lingual embeddings in our experiment.

We believe that code-mixed word embeddings have good potential also for other NLP tasks that require cross-lingual word embeddings. If we could use code-mixed word embeddings, the time and cost needed to be spent could be reduced, since code-mixed word embeddings do not require a sentence-aligned corpus or the process to align monolingual word embeddings. It does require either a large code-mixed corpus, or a synthesized corpus, however, as we have shown, good quality code-mixed embeddings can be had also with a simple and resource-lean synthesis method. A further potential advantage of code-mixed embeddings is that they model code-mixed words in the context of both languages, which might be advantageous.

Result in percentage		Mean (STD)	Best
Baseline Embedding			
1	English	62.60 (1.94)	64.77
2	Indonesia	62.87 (2.45)	64.77
Cross Lingual Embedding			
3	Supervised	63.05 (1.60)	66.82
4	Semi-supervised	63.33 (1.49)	65.53
5	Identical	63.17 (1.84)	67.27
6	Unsupervised	63.33 (2.54)	66.36
Code-Mixed Embedding			
7	Code-mixed	63.42 (1.65)	67.27

Table 4: List of sentiment classification cross-validation score employing different embedding

Word Embedding	OOV	Type
Cross-Lingual	6,206	148,737
Code-Mixed	6,406	92,867

Table 5: Number of Out-of-Vocabulary (OOV) on word embedding. The type is the number of distinct word representation in the word embedding used in sentiment classification

9. Bibliographical References

Adriani, M., Asian, J., Nazief, B., Tahaghoghi, S. M., and Williams, H. E. (2007). Stemming indonesian: A confix-stripping approach. *ACM Transactions on Asian Language Information Processing (TALIP)*, 6(4):1–33.

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

Barik, A. M., Mahendra, R., and Adriani, M. (2019). Normalization of Indonesian-English code-mixed twitter data. In *Proceedings of the 5th Workshop on Noisy User-generated Text (W-NUT 2019)*, pages 417–424.

Barman, U., Das, A., Wagner, J., and Foster, J. (2014). Code mixing: A challenge for language identification in the language of social media. In *Proceedings of the first workshop on computational approaches to code switching*, pages 13–23.

Belazi, H. M., Rubin, E. J., and Toribio, A. J. (1994). Code switching and x-bar theory: The functional head constraint. *Linguistic inquiry*, pages 221–237.

Brown, H. D. (2000). *Principles of language learning and teaching*. Longman, New York.

Cárdenas-Claros, M. S. and Isharyanti, N. (2009). Code-switching and code-mixing in internet chatting: Between'yes," ya,'and'si'-a case study. *The Jalt Call Journal*, 5(3):67–78.

Çetinoğlu, Ö. (2016). A Turkish-German code-switching corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4215–4220.

Cettolo, M., Girardi, C., and Federico, M. (2012). WIT^3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May.

Chakma, K. and Das, A. (2016). Cmir: A corpus for evaluation of code mixed information retrieval of Hindi–English tweets. *Computación y Sistemas*, 20(3):425–434.

Chang, C.-T., Chuang, S.-P., and Lee, H.-Y. (2019). Code-switching sentence generation by generative adversarial networks and its application to data augmentation. In *Twentieth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 554–558. International Speech Communication Association.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46.

Danet, B. and Herring, S. C. (2003). Introduction: The multilingual internet. *Journal of Computer-Mediated Communication*, 9(1):JCMC9110.

Di Sciullo, A.-M., Muysken, P., and Singh, R. (1986). Government and code-mixing. *Journal of linguistics*, 22(1):1–24.

Gambäck, B. and Das, A. (2014). On measuring the complexity of code-mixing. In *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*, pages 1–7.

Habib, S. T. (2014). Code mixing in Twitter among students of English studies 2010 at Universitas Indonesia. In *Makalah Non-Seminar*.

Hoffmann, C. (2014). *Introduction to bilingualism*. Routledge.

Iyyer, M., Manjunatha, V., Boyd-Graber, J., and Daumé III, H. (2015). Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1681–1691, Beijing, China, July. Association for Computational Linguistics.

Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., and Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Kramarae, C. (1999). The language and nature of the internet: The meaning of global. *New media & society*, 1(1):47–53.

Kurniawan, B. (2016). Code-mixing on Facebook postings by EFL students: A small scale study at an SMP in Tangerang. *Indonesian JELT*, 11(2):169–180.

Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

Lee, G., Yue, X., and Li, H. (2019). Linguistically motivated parallel data augmentation for code-switch language modeling. In *Twentieth Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 3730–3734. International Speech Communication Association.

Lison, P., Tiedemann, J., Kouylekov, M., et al. (2018). OpenSubtitles2018. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC 2018)*. European Language Resources Association (ELRA).

Marzona, Y. (2017). The use of code mixing between Indonesian and English in Indonesian advertisement of Gadis. *Jurnal Ilmiah Langue and Parole*, 1(1):238–248.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Myers-Scotton, C. (1997). *Duelling languages: Grammatical structure in codeswitching*. Oxford University Press.

Poplack, S. (1980). Sometimes I'll start a sentence in Spanish y termino en espanol: toward a typology of code-switching1. *Linguistics*, 18(7-8):581–618.

Poppi, F. (2014). Global interactions in English as a lingua franca. how written communication is changing under the influence of electronic media and new contexts of use. *Ibérica*, 28:225–256.

Pratapa, A., Bhat, G., Choudhury, M., Sitaram, S., Dandapat, S., and Bali, K. (2018a). Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553, Melbourne, Australia, July. Association for Computational Linguistics.

Pratapa, A., Choudhury, M., and Sitaram, S. (2018b). Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium, October-November. Association for Computational Linguistics.

Purwarianti, A., Andhika, A., Wicaksono, A. F., Afif, I., and Ferdian, F. (2016). InaNLP: Indonesia natural language processing toolkit, case study: Complaint tweet classification. In *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, pages 1–5. IEEE.

Ruder, S., Vulić, I., and Søgaard, A. (2017). A survey of cross-lingual word embedding models. *arXiv preprint arXiv:1706.04902*.

Samih, Y. and Maier, W. (2016). An Arabic-Moroccan Darija code-switched corpus. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4170–4175.

Sankoff, D. (1998). code-switching. *Bilingualism, Language and Cognition*, 50:39–50.

Saputri, M. S., Mahendra, R., and Adriani, M. (2018). Emotion classification on Indonesian twitter dataset. In *2018 International Conference on Asian Language Processing (IALP)*, pages 90–95. IEEE.

Setiawan, D. (2016). English code switching in Indonesian language. *Universal Journal of Educational Research*, 4(7):1545–1552.

Shafie, L. A. and Nayan, S. (2013). Languages, code-switching practice and primary functions of Facebook among university students. *Study in English Language Teaching*, 1(1):187–199.

Sharma, S., Srinivas, P., and Balabantaray, R. C. (2015). Text normalization of code mix and sentiment analysis. In *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1468–1473. IEEE.

Siregar, M., Bahri, S., Sanjaya, D., et al. (2014). Code switching and code mixing in Indonesia: Study in sociolinguistics. *English Language and Literature Studies*, 4(1):77–92.

Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In Nicoletta Calzolari (Conference Chair), et al., editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Vijay, D., Bohra, A., Singh, V., Akhtar, S. S., and Shrivastava, M. (2018). Corpus creation and emotion prediction for Hindi–English code-mixed social media text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 128–135.

Vilares, D., Alonso, M. A., and Gómez-Rodríguez, C. (2016). En-es-cs: An English–Spanish code-switching twitter corpus for multilingual sentiment analysis. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 4149–4153.

Wick, M., Kanani, P., and Pocock, A. (2016). Minimally-constrained multilingual embeddings via artificial code-switching. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Winata, G. I., Madotto, A., Wu, C.-S., and Fung, P. (2019). Code-switched language models using neural based synthetic data from parallel sentences. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 271–280, Hong Kong, China, November. Association for Computational Linguistics.

10. Language Resource References

Tiedemann, J. (2012). *OPUS ... the open parallel corpus*. `http://opus.nlpl.eu/`.