École Polytechnique Fédérale de Lausanne

Master Thesis in Computer Science

Creation and Evaluation of Multilingual Embeddings for Downstream Tasks

Author:
Meryem M'hamdi[1] meryem.mhamdi.@epfl.ch

Supervisors:
Dr. Claudiu Musat [2] Claudiu.Musat@swisscom.com
Prof. Robert West [3] robert.west@epfl.ch

Lausanne, August 10, 2018

[1] Computer Science, EPFL, Switzerland
[2] Research Director, Artificial Intelligence Machine Learning Group, Swisscom, Switzerland
[3] Assistant professor and Director of Data Science Lab, School of Computer and Communication Sciences at EPFL, Switzerland

Abstract

Multilingual Embeddings are an extension of word embeddings aiming at bridging the gap between different languages by learning a way to align them into a shared vector space. Several models have been proposed for inducing them, requiring different levels of supervision and training mechanisms. Their ability to learn a common representation of words across languages makes them attractive to numerous intrinsic and extrinsic NLP applications. For example, some popular applications include cross-lingual word similarity and translation, transfer learning in document classification and syntactic parsing which are used as benchmarks for evaluating their performance but which often give mixed results. Several studies have focused more on devising new methodologies for learning the embeddings; however, an extensive evaluation of their gain on performance compared to their monolingual counterpart using state-of-the-art methodologies for different applications is lacking. Moreover, it is not clear how changing the application could impact their feasibility.

In this thesis, we start by reproducing some state-of-the-art methodologies for creating multilingual embeddings for four languages: English, German, French and Italian. We focus on fine-tuned models due to their efficiency and obtain some models already trained from scratch for comparison. We directly integrate the learned embeddings into a pipeline of several downstream applications namely document classification and churn detection. We follow state-of-the-art methodology for each task leveraging deep learning models mainly a combination of CNN and GRU and hierarchical attention networks. In a second experiment, we jointly train the embeddings and the document classification to test the impact of specializing the embeddings to the current task and deduce how well the embeddings can do by themselves. We define a new pipeline for performing unsupervised event detection which benefits from multilingual embeddings and different metrics for evaluating their performance and apply this approach to 11 different languages. In all experiments, we compare the performance of multilingual training with multilingual embeddings to its monolingual counterpart.

KEYWORDS: Distributed representations, multilingual learning, Deep Learning, Cross-Lingual Document Classification, Cross-Lingual Churn Detection, Cross-Lingual Event Detection, Natural Language Processing

Acknowledgements

I would like to express my gratitude to my supervisors: Dr. Claudiu Musat and Professor Dr. Robert West for providing me with this wonderful opportunity to work on such an inspiring project. I have learned a lot thanks to their patient guidance, enriching discussions, enthusiastic encouragement and endless support.

My special thanks go to Christian Abbet who worked on providing the datasets for churn detection and with whom I collaborated to publish one paper on multilingual churn detection. This publication could not be possible without the very helpful and kind feedback and review of Athanasios Giannakopoulos, Dr. Andreea Hossmann, and Dr. Michael Baeriswyl.

I also would like to extend my thanks to all my Swisscom colleagues and lab mates for making this journey an amazing experience. Thank you my parents for your consistent support and your unconditional love and confidence without which I would not be able to go that far.

Contents

List of Figures

List of Tables

List of Abbreviations

CCA	Canonical Correlation Analysis
MLP	Multi-layer Perceptron
FT-MLP	Fine Tuned Multi-layer Perceptron
CNN	Convolutional Neural Networks
MF-CNN	Multi-Filter Convolutional Neural Networks
bi-GRU-Att	Bidirectional Gated-Recurrent Units with Attention
CNN-biGRU-Att	Cascaded CNN and bi-GRU-Att
GRU	Gated-Recurrent Unit
LSTM	Long-Short Term Memory
HAN	Hierarchical Attention Networks
CLDC	Cross-Lingual Document Classification
CLCD	Cross-Lingual Churn Intent Detection
CLED	Cross-Lingual Event Detection
RCV1/RCV2	Reuters Corpus Volume I/II

1 Introduction

Distributional word embeddings have become an important technique and a crucial game changer in natural language processing. Unlike other representations such as one hot encoding and TF-IDF, these vectors can efficiently map words into a low dimensional space such that words are closer to each other are the ones which also exhibit similar syntactic and semantic characteristics according to the distributional hypothesis Harris (1954). Since they can be learned efficiently from large unsupervised corpora, especially with the adoption of techniques like negative and noise contrastive estimation in Mikolov et al. (2013a) and Mikolov et al. (2013b), word embeddings have become ubiquitous in numerous applications not restricted to NLP. This goes beyond computing similarities between words and includes semantic understanding tasks namely sentiment analysis Maas et al. (2011), parsing Socher et al. (2013), document summarizing Wang et al. (2016) to name a few. It also has numerous applications in information retrieval, recommendation systems, and dialogue management systems Yan et al. (2016).

Multilingual embeddings are a natural extension of traditional embeddings where the aim is to learn representations for multiple languages at the same time. Embeddings that can generalize well across languages are highly desirable given the scarcity of annotated data in some languages. Training several language-specific models is usually costly and highly depends on the quality of annotation for each language independently. Resorting to machine translation is not the optimal solution as it requires money and time in addition to the fact that errors propagated throughout the process may seriously affect the accuracy of the application at hand.

Swisscom Digital Lab is concerned about research projects in Natural Language Processing and Speech Processing aiming at improving current technological solutions and opening new opportunities for their clients. Given the multilingual environment and nature of life and interaction between customers in Switzerland with four official languages, conceiving solutions that go beyond one language is one of the primary long-term objectives of the company. One of the primary applications that can benefit from thid multilingual capability is multilingual chat-bot which can understand and interact with customers speaking different languages. Another universal application is the extraction of relevant information from text to detect main events and any anomalies indicating customers wishing to leave the company. The objective of this thesis is to evaluate and propose an adequate and efficient solution relying on multilingual embeddings that can make use of multiple languages at the same time.

Different Methodologies for inducing multilingual embeddings have been the subject of many recent papers. These embeddings can either be trained from scratch where monolingual and multilingual constraints are optimized at the same time (Klementiev et al. (2012), Gouws et al. (2015), Luong et al. (2015)) or fine-tuned on top of monolingual embeddings (Smith et al. (2017), Mrksic et al. (2017)). Some of those methodologies train the embeddings before applying them to the task (Ammar et al. (2016)), others train them jointly with the task (Ferreira et al. (2016), Wang et al. (2017), Zhou et al. (2015)). Training the embeddings can rely on word, sentence, document alignment, a hierarchical combination of document and sentence representation Hermann and Blunsom (2014) or no alignment at all Conneau et al. (2017).

To evaluate the obtained embeddings, previous work defined several benchmarks which are multilingual versions of monolingual embeddings direct applications mainly multilingual word similarity and word translation. The quality of those embeddings is also tested using downstream applications notably cross-lingual document classification which was introduced by Klementiev et al. (2012) and more recently sentiment classification Zhou et al. (2015) and multilingual chatbot Mrksic et al. (2017). However, a comparison between training using monolingual vs. multilingual embeddings is not shown, and the gain on performance when using multilingual embeddings to train a model on any language subset is not clear. In addition to that, most models used are outdated by now, so it is not straightforward which architectures multilingual embeddings lead to better gain on performance. Moreover, since this is a relatively new research trend, there are still many possible applications of multilingual embeddings that have not been explored.

In this thesis, we survey several methodologies for inducing multilingual embeddings and investigate the ups and downs in performance with respect to downstream applications and analyze the impact of the chosen models for each application. We focus on three main tasks first of which concerns **Cross-Lingual Document Classification** which is a reproduction of a famous benchmark used in literature, while the second application is **multilingual churn detection** is another yet a new variation of text classification and propose a new pipeline that makes use of multilingual embeddings for a better **language agnostic event detection**.

In the first part, we start by creating multilingual embeddings. We not only focus on fine-tuned models due to their efficiency but also obtain some models already trained from scratch for comparison. We try offline methodologies for fine-tuning the embeddings using ground truth translation pairs learned using SVD, and we reproduce unsupervised embeddings where the embeddings are trained through iterative optimization. Our aim is to find which embeddings perform better under each application and choice of model.

In the second and third parts, we evaluate two ways of integrating the resulting embeddings into our two variations of text classification: document classification and churn detection model pipelines. We follow state-of-the-art methodologies for each task leveraging deep learning models mainly a combination of CNN and GRU, and Hierarchical Attention Networks. In a second experiment, we jointly train the embeddings and the document classification to test the impact of specializing the embeddings to the current task and deduce how well the embeddings can do by themselves. In all experiments, we compare the performance of multilingual training with multilingual embeddings to its monolingual counterpart.

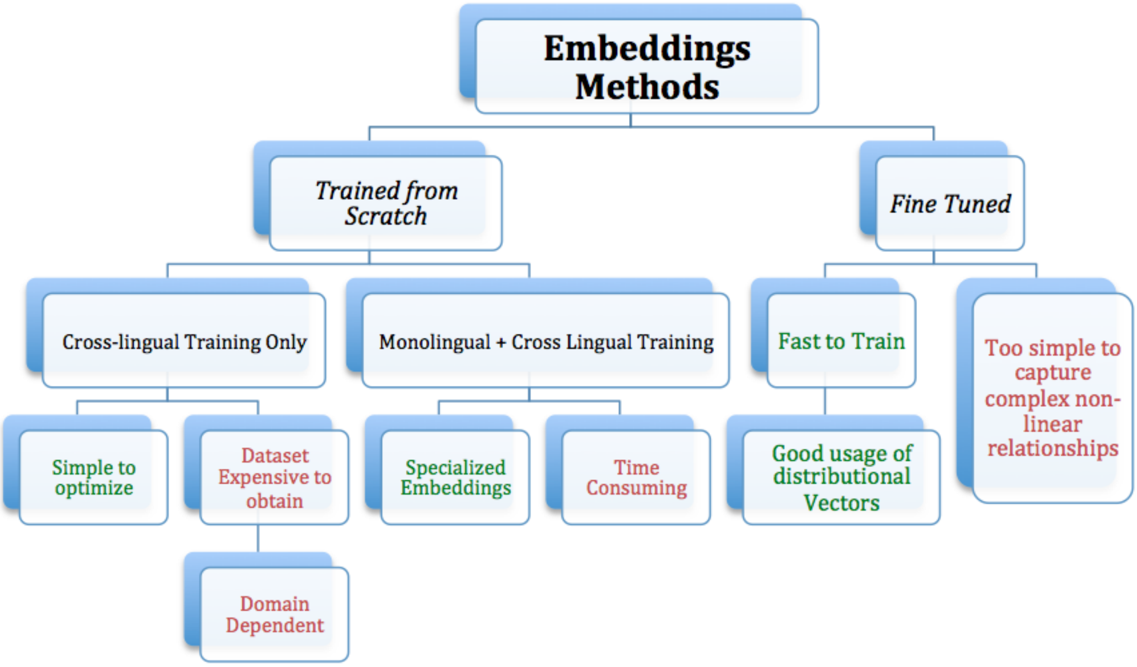
In the fourth part, we define a new pipeline for performing unsupervised event detection which benefits from multilingual embeddings. Since we aim at analyzing events in a new context that has never been analyzed before and due to the lack of a large-scale multilingual annotated dataset for event detection, we follow an unsupervised approach to detect essential events in Twitter streams. We take World Cup 2018 manifestation as our use case. Since tweets about such an international worldwide manifestation cover a large set of languages, we test the feasibility of a multilingual approach to extract related events as efficiently and accurately as possible. In other words, instead of considering tweets in each language separately, we take the aggregation of tweets in a set of representative languages to train a robust model for event detection. We define an event as a bursty abnormality which translates into active communication as different categories of people either participating or witnessing the event form to support diverging outcomes of the event. We consider keyword burstiness to select candidate triggers that we map into a commonly shared embedding space using multilingual embeddings before clustering them into events. In the end, we define certain metrics for evaluating their performance qualitatively and quantitatively against a ground truth of some fine-grained events within individual games.

2 Literature Study

The primary goals of this thesis are to evaluate the performance of multilingual embeddings, investigate the types of downstream applications for which their use brings value both in terms of accuracy and efficiency and which kind of approaches for inducing multilingual embeddings work best for each application. In an attempt to have a better understanding of multilingual embeddings, we focus in this thesis on evaluating them against three tasks that share some similarities in the way the embeddings can be integrated into their pipeline to ensure a smooth extension from monolingual to multilingual applicability. These tasks include: cross-lingual document classification, churn intent detection and event detection.

Before surveying previous work for each task separately, we summarize the main findings in literature research regarding the creation of multilingual embeddings, which is the basis of our comparative analysis. Due to the similarities between the first two tasks, we group them into the category of text classification. Since multilingual embeddings were never used for event detection, we include previous work covering other methodologies that work across languages.

2.1 Inducing Multilingual Embeddings



Embeddings Methods
Trained from Scratch
Fine Tuned
Cross-lingual Training Only
Monolingual + Cross Lingual Training
Fast to Train
Too simple to capture complex non-linear relationships
Simple to optimize
Dataset Expensive to obtain
Specialized Embeddings
Time Consuming
Good usage of distributional Vectors
Domain Dependent

Figure 2.1: Multilingual Embeddings Different Methodologies Tradeoffs

Monolingual word embeddings (Mikolov et al., 2013b) have become ubiquitous to almost all NLP applications. By relying on word co-occurrence and distributional approaches, they are made able to capture crucial linguistic characteristics by mapping words into the same low dimensional space. Multilingual embeddings are an extension of word embeddings to multiple languages. If monolingual word embeddings can learn useful features in a specific language, cross-lingual word embeddings aim at learning features which capture the similarities between words belonging to different languages at the same time. Existing approaches for creating such embeddings can be classified based on the training mode into two main families: training from scratch and fine-tuning. Figure 2.1 summarizes the different types of embedding methodologies exposing their pros and cons.

2.1.1 Training from Scratch

Cross-lingual word embeddings can be learned from scratch, without the use of any prior distributional monolingual information, optimizing for cross-lingual alignment or training monolingual constraints alongside with cross-lingual constraints:

Optimization of Bilingual Constraints Only Some approaches only take into consideration the bilingual constraint such as in (Chandar et al., 2014) which uses an auto-encoder to reconstruct the bag of words for aligned sentences without requiring word level alignment. Luong et al. (2015) extends the skip-gram model initially proposed by Mikolov et al. (2013a), by predicting not only the context of a word in the source language, but also by predicting the context in a target language of a word in the source language. This makes use of only parallel sentences while word alignments can be estimated using the alignments of its neighbors and by assuming sentences are monotonically aligned. While efficient techniques were devised in this line of research which makes models relying solely on parallel data fast to train, the resulting embeddings are usually biased to the domain of the parallel data used to train them. This parallel data is not abundant enough especially for low-resource languages and do not necessarily belong to the same domain they are going to be applied to.

Joint Optimization of Monolingual and Bilingual Constraints This research direction was initiated with the work of Klementiev et al. (2012) which follows a multi-task approach to induce and align embeddings for multiple languages at the same time. Neural language models are trained for each language and jointly optimized using a cross-lingual objective based on word alignment obtained from machine translation. However, as this model was developed before Mikolov et al. (2013a), it didn't make use of the new optimization techniques such as negative sampling (NS) and noise-contrastive estimation (NCE). Gouws et al. (2015) exploited both works to build BilBOWA, a faster approach that optimizes jointly for monolingual and bilingual constraints and that can scale to large monolingual datasets. Unlike Klementiev et al. (2012) who use word alignment obtained from translation, Gouws et al. (2015) train multi skip-gram assuming uniform alignment between sentences where each word in a source sentence is aligned with every word in the target sentence. Although optimizing both monolingual and cross-lingual objectives is made more efficient and can lead to less biased embeddings as it allows for training on any available monolingual data, its performance largely depends on the task and languages on which it is evaluated. In general, this kind of models tend to perform well for tasks like Cross-Lingual Document Classification and Cross-Lingual Dictionary Induction as shown in (Upadhyay et al., 2016).

2.1.2 Fine Tuned Models

Instead of training from scratch, some models make use of existing distributional vectors trained on each language independently and learn at a later stage the cross-lingual alignment by fine-tuning them to the bilingual constraints. For example, Faruqui and Dyer (2014) use canonical correlation analysis based on a bilingual dictionary to learn linear projection between distributional vectors obtained separately for each language. In other words, the projection in both directions is selected in such a way to maximize the correlation between the bilingual dictionary pair entries. Another variation of linear projection uses SVD to learn directly and efficiently bilingual word embeddings (Smith et al., 2017).

Other studies like (Conneau et al., 2017) propose unsupervised methodologies using Generative Adversarial Networks to alleviate the need for dictionary pairs. Unfortunately, their performance is not as good as their supervised counterparts and they are only best on pairs of languages sharing the alphabets. This makes them not interesting to investigate in our case since we aim at exploiting a methodology which can keep the same level of performance across languages. For these reasons, we take into consideration only supervised methodologies.

Other works like (Faruqui and Dyer, 2014) inject semantic knowledge to fine-tune those embeddings. Their methodology termed as "retrofitting" aims at coming up with vector representations such that synonyms are closer together. dblp-mrksic-17 use not only synonymy but also antonymy constraints from different languages and show that initializing the embeddings using distributional vectors facilitates semantic transfer between languages. In general, the performance of this family of models is on par with models that jointly optimize for monolingual and bilingual constraints especially when strong levels of supervision are used.

2.2 Multilingual Embeddings Applications

Previous work inducing multilingual embeddings evaluate the quality of their methodologies against a different range of applications. Those can be categorized into either intrinsic or extrinsic as shown in the figure 2.2. The quality of multilingual embeddings is either evaluated intrinsically by directly testing their ability to capture syntactic and semantic relationships between words. Such benchmarks include word similarity, word translation, and correlation based evaluation. Extrinsically, those multilingual models are evaluated on their performance when used as input features to downstream semantic transfer tasks. In the remaining of this section, we focus in particular on previous work related to cross-lingual embeddings applications to text classification and event detection.

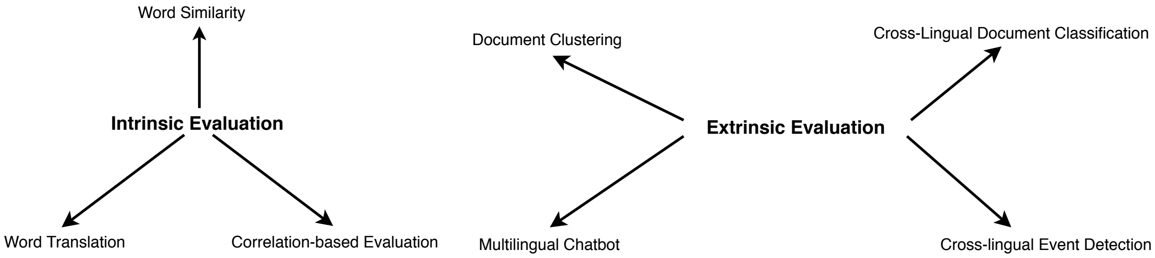


Figure 2.2: Multilingual Embeddings Applications and Evaluation Tasks

2.2.1 Cross-lingual Text Classification

Multilingual word embeddings were mainly applied in the context of tasks like Cross-Lingual Document Classification benchmark (CLDC) initially defined in (Klementiev et al., 2012). To evaluate the quality of the induced multilingual embeddings, they test their ability to act as agents in the direct transfer learning. More specifically, they train a model on labeled documents in a source language and apply it directly to classify unlabeled documents in a target language. However, a comparison between the performance using monolingual vs. multilingual data is missing.

Other work evaluated multilingual embeddings on tasks like Cross Language Sentiment Classification (CLSC) as in (Zhou et al., 2015). The idea is to jointly train bilingual embeddings using the documents annotated with sentiments and their translations to other languages and show that the multilingual approach outperforms monolingual experiments. This gain in performance is what encouraged us to explore this methodology on churn intent detection which is a variation of text classification on short text. To the best of our knowledge, there is no prior work adopting a multilingual approach either using multilingual embeddings or else for churn intent detection.

2.2.2 Event Detection in Social Media

Due to the lack of previous work on multilingual event detection in social media, we also include work concerning event detection in general. Event detection in social media can be carried out either in a supervised, unsupervised or hybrid manner. One recent work which followed a supervised approach such as (Nguyen and Grishman, 2015) uses Convolutional Neural Networks to avoid the trouble associated with complicated feature engineering and error propagation from external resources. While this approach serves to automatically extract relevant features, it relies on the annotation of large-scale datasets which is too costly, restricted to one domain and not adapted to real-time event detection.

For those reasons, other work considered either semi-supervised or unsupervised methodology which is usually based on the clustering of features identified as triggers. There are different frameworks for feature extraction for this line of research. While some previous works rely on temporal or spatial frequencies to measure keyword importance, other works rely on its emotional score. For instance, (Valkanas and Gunopulos, 2013) focus on sentiment level fluctuations assuming that the more the emotional state is affected, the more likely some event occurred. They apply those sentiment sensors to tweets that were initially clustered by geographic location so that the sentiment of each region is analyzed. Some use textual features solely, whereas others use a combination of both textual and non-textual features. For instance, (Abdelhaq et al., 2013) relies on the initial clustering of keywords based on spatial signatures calculated using geo-referenced tweets. Scoring of keywords is based on their level of burstiness, spatial distribution and other time-related features. Each cluster receives a score equal to the sum of its keywords' score. Relying on the same principle of burstiness, (Li et al., 2012) create the Twevent system which is more adapted to social media. Before delving into burstiness distribution analysis, they start by detecting proper segmentation using Wikipedia resources. Then, they not only incorporate tweet frequency but also user frequency in the way burstiness score is computed. After that, they cluster bursty segments to detect events before applying the second round of filtering based on segment newsworthiness score.

Most of the above use TF-IDF as a mechanism for feature representation. On the other hand, the use of word embeddings as feature extractors in event detection is not that common except the work of (Ertugrul et al., 2017). After embedding message terms extracted from Turkish tweets into a continuous space, they group them using hierarchical clustering. By evaluating their approach against four different unexpected and scheduled events, they show that utilizing word embeddings for event detection outperform TF-IDF based vector representations.

Compared to event detection in social media conducted on one language, multilingual event detection using textual features has not gained that much attention due to the complexity of the task. Nonetheless, some attempts have been made for a language-agnostic event detection based on non-textual features (Buntain, 2014). Instead of building expensive pipelines for each language independently, they propose to make use of temporal characteristics of tokens in either English or Spanish, build a classifier to detect bursty tokens, then relate them to known sport events. They also use transfer learning to generalize on unseen types of sports events. Although there is some research attempting to address multilingual event detection based on non-textual features, this line of research focuses only on burstiness and needs a large number of training data. To the best of our knowledge, no prior work explores multilingual embeddings and its ability to train efficient and performant models for multilingual event detection.

2.3 Multi-Tasking Embeddings with the Application

While most works rely on learning first the embeddings then applying them directly to the task of interest, some work design methodologies to train the embeddings alongside the application using different forms of multi-tasking. This has been employed in the context of numerous applications mainly sentiment analysis, named entity recognition and document classification. Zhou et al. (2015) propose a methodology to learn a cross-lingual representation of sentiment information to enable sentiment classification. Wang et al. (2017) multi-task training bilingual word embedding with named entity recognition. For cross-lingual document classification, Ferreira et al. (2016) propose a model that jointly learns to embed and predict multilingual documents by optimizing for a loss that combines cross-lingual training loss with supervised document classification loss. Despite the simplicity of each loss component relying on cross-lingual multilingual embeddings regularization vs. joint cross-lingual and monolingual embeddings and logistic loss for document classification, this model manages to surpass most other state-of-the-art models. This motivates us to investigate a multi-tasking model where a more complex model is adapted for document classification.

3

Creation of Multilingual Embeddings

3.1 Overview

In this chapter, we describe the different methodologies used to generate multilingual embeddings that have been employed in the evaluation of the applications of our interest in this thesis. The choice of the embeddings to be evaluated is picked carefully to comply with the previous findings in literature proving that models requiring higher levels of supervision perform best and at the same time to make smart use of existing monolingual embeddings and enable easy comparison between monolingual and multilingual settings. It should be noted that this study is not comprehensive in terms of the covered embeddings as it is impossible to investigate all methodologies.

Therefore, we pick some representative variations from the two families: fine-tuned and trained from scratch. Fine-tuned models include offline methodologies using either SVD or CCA for learning the alignment on top of monolingual embeddings, and another approach making use of semantic information known as Attract-Repel. We take two instances of models trained from scratch namely cross-lingual training using sentence alignment and joint monolingual and cross-lingual training using skip-gram. We also try specializing them by jointly fine-tuning with the classification task and multi-tasking their creation with the classification task which we explain later in sections 4.3.2.1 and 4.3.2.3 respectively.

3.2 Fine Tuned Methodologies

3.2.1 Linear Transformation of Monolingual Embeddings

We try to build multilingual embeddings which map words from different languages into one joint vector space by learning translations of monolingual embeddings into a target space. We set English as the target space and we learn the transformation matrix that aligns other languages to English using bilingual translation pairs. In other words, this approach fine-tunes non-English embedding by applying a linear transformation that maps them into the English space. This offline methodology has the advantage of guarantying a fair comparison between the performance of monolingual and multilingual settings and shows clearly the added value of multilingual approach when both monolingual and multilingual embeddings are trained on the same monolingual corpora.

Monolingual Embeddings We use 300-dimensional FastText pre-trained monolingual word embeddings obtained and introduced in (Bojanowski et al., 2017)[1]. Those are not only used for building multilingual embeddings (based on linear projection) but are also the same word embeddings used for monolingual experiments. Traditionally, this is solved using stochastic gradient descent to learn a linear transformation from the source to target embeddings space by minimizing the reconstruction error. Alternatively, there exist efficient ways for obtaining them directly including Singular Value Decomposition (SVD) and Canonical Correlation Analysis (CCA).

[1] A large repository of pre-trained embeddings can be obtained from https://github.com/facebookresearch/FastText/blob/master/pretrained-vectors.md

For simplicity, we describe in what follows the approach used to generate bilingual embeddings for English EN and German DE. Including more languages is straightforward as it suffices to learn an alignment for each non-English language $l \in L \setminus \{EN\}$ to the English space. English is used as the target multilingual space, as it has the richest vocabulary and is trained on the largest corpora and also due to the availability of rich bilingual dictionaries involving English on one side.

3.2.1.1 Using SVD (multi(pseudo_dict), multi(exp_dict))

Method We follow the same approach described in (Smith et al., 2017). We learn the alignment on top of monolingual embeddings using the training split of the expert bilingual dictionary. This provides a regularization term which tunes the initial monolingual embeddings by pushing words sharing the same meaning provided by the dictionary to be closer to each other in the new joint vector space. Therefore, the problem of building bilingual embeddings reduces to learning the linear transformation matrix $W_{DE \rightarrow EN}$ which maps the source German monolingual space into the English space as illustrated in figure 3.1.

Formally, given X and Y monolingual word vector matrices for the source and target spaces (in our case German and English respectively), the goal is to learn W that maximizes the cosine similarity defined by:

$$\max_W \sum_{i=1}^{n} y_i^T W x_i \tag{3.1}$$

Smith et al. (2017) proves that this transformation needs to be orthogonal in order to ensure that we are not only able to map the source language into the target language but also the target language back into the source. Then, the optimization objective learns the orthogonal matrix O as follows:

$$\max_O \sum_{i=1}^{n} y_i^T O x_i \text{ subject to } O^T O = I \tag{3.2}$$

such that $x_i \in X_D$ and $y_i \in \mathrm{Y}_D$ where X_D and Y_D are source and target word vectors in the paired bilingual dictionary. This optimization objective can be solved directly and efficiently using SVD of the product of the paired dictionary matrices [2]:

$$\mathrm{M} = \mathrm{Y}_D^T X_D = U \sum V^T \tag{3.3}$$

The resulting U and V vectors are orthonormal matrices whose product gives us the desired transformation matrix W. We also apply dimensionality reduction by keeping only the first rows in matrices U and V which correspond to large values in the diagonal matrix $\sum$. We experiment with two kinds of embeddings depending on the type of dictionary pairs used to train them:

- *pseudo_dict*: we exploit identical strings across languages to construct dictionary pairs. For example, words like "Paris", "DNA" are universal and are the same for all languages
- *expert_dict*: we use ground truth bilingual dictionaries (Conneau et al., 2017)[3] consisting of translation pairs for each pair of source and target languages (where the target language is always English). Only the train split (consisting of 5000 pairs) is used for training while 1500 pairs are used for testing the quality of the embeddings before feeding them to the downstream applications.

3.2.1.2 Using CCA (multi(CCA))

We follow the same multilingual methodology [5] described in (Ammar et al., 2016) which builds upon the bilingual approach of (Faruqui and Dyer, 2014). Unlike SVD, not only one but two linear mappings are learned: from the source to the shared multilingual space $W_{DE} \rightarrow W^*$ and from the target to the shared multilingual space $W_{EN} \rightarrow W^*$. Those projections are obtained by maximizing the correlation between $W_{DE} \rightarrow W^* \times X_D$ and $W_{EN} \rightarrow W^* \times Y_D$ using bilingual dictionaries. Then, the transformation of German to the multilingual space is obtained using $X_{new} = W_{EN}^{-1} \rightarrow W^*$

[2] Please refer to appendices of (Smith et al., 2017) for the full proof

[3] A large repository of up to 110 bilingual dictionaries covering high and low resource languages is available in https://github.com/facebookresearch/MUSE

[4] Figure outline inspired and adapted from (Conneau et al., 2017)

[5] Code and Pre-trained multi-CCA embeddings for up to 59 languages were obtained from http://128.2.220.95/multilingual/data/. We use 512-dimensional embeddings trained on 13 languages for CLDC and CLCD and 59 languages for Event Detection

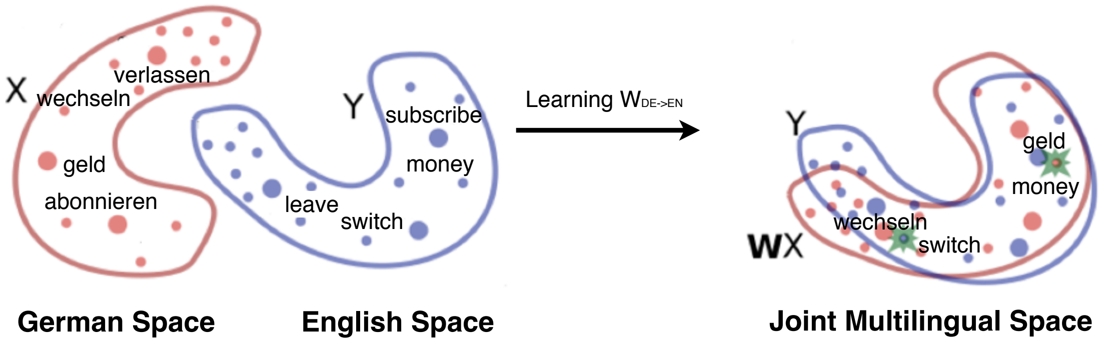


Figure 3.1: Multilingual Alignment of Word Embeddings [4]

$\times \mathrm{W}_{DE\rightarrow W^{*}\times X}$ and English embeddings are kept as they are as they are the chosen target multilingual space.

3.2.2 Attract-Repel (multi(sem))

We follow the methodology[6] of Mrksic et al. (2017) for generating semantically specialized multilingual embeddings aka Attract-Repel [7]. The idea is to use monolingual and cross-lingual synonyms and antonyms to inject linguistic constraints to distributional monolingual vectors. The problem is thus reduced to learning embeddings that are close to or far away from each other in the space in case of synonymy and antonymy respectively. Mrksic et al. (2017) define and implement a methodology that operates over mini-batches of synonyms B_S and antonyms B_A across languages and consider their negative examples T_S for synonymy T_A for antonymy using negative sampling. In the end, the cost function that is optimized using adaGrad algorithm consists of the sum of all synonymy S and antonymy A regularizers as follows:

$$\mathrm{C}(\mathrm{B}_S, T_S, B_A, T_A) = S(B_S, T_S) + A(B_A, T_A) + R(B_S, B_A) \tag{3.4}$$

Bilingual to Multilingual Embeddings In order to create the alignment from bilingual to multilingual, we learn the weights for two linear projections: from EN-FR to EN-DE and from EN-IT to EN-DE to bring the French part of EN-FR and Italian part of EN-IT to the same joint space as EN-DE as illustrated in figure 3.2. We solve each linear projection using logistic regression optimized using stochastic gradient descent.

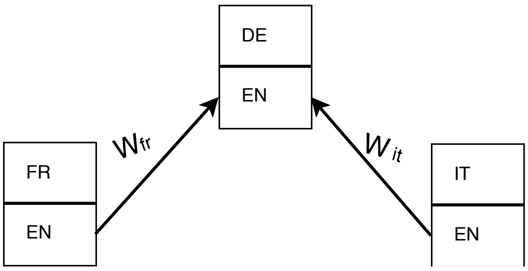


Figure 3.2: Bilingual to Multilingual

For illustration, we describe the approach for learning the mapping $W^{*} = W_{EN-FR\rightarrow EN-DE}$. The idea is to make use of the inherent parallelism between the two spaces in the sense that English vectors for words in space EN-FR should be aligned to vectors of the same words in space EN-DE. Formally, let u_i and v_i be the vectors of word i in space EN-FR and EN-DE respectively. So, we

[6] The vectors can be obtained from https://github.com/nmrksic/attract-repel
[7] Code is available in www.github.com/nmrksic/attract-repel

learn the matrix W^* such that $u_i \simeq W^* v_i$. This can be solved by minimizing the Euclidean distance between English words shared between the two spaces as follows:

$$\textstyle\sum_{i=1}^{n} \parallel u_i - W^* \cdot v_i \parallel^2 = \parallel U - W^* \cdot V \parallel^2 \tag{3.5}$$

where U and V are embeddings matrices where each row corresponds to vector in EN-FR and EN-DE of each word shared between the two spaces and $\parallel . \parallel_F$ is the Frobenius norm.

To solve this m-variate linear regression, we use stochastic gradient descent (SGD) which is solved using Vowpal Wabbit, a library that can handle large-scale data efficiently. To comply with VW inability to deal with multidimensional output, we split the problem to single output linear regression sub-problems. Therefore, for each sub-problem, we create a VW file for each embeddings dimension $j = [1, 2, .., n]$ of u. The format of the file looks like:

$$u_{1j}|1 : v_{11} 2 : v_{12} ... n : v_{1m}$$

$$u_{2j}|1 : v_{21} 2 : v_{22} ... n : v_{2m}$$

$$u_{nj}|1 : v_{n1} 2 : v_{n2} ... n : v_{nm}$$

where n is the number of words, m is the dimensionality of the embeddings. Running optimization for this file results in the j^{th} column of the desired transformation W^*. In the end, this transformation is applied to French vectors in EN-FR (and Italian vectors in EN-IT with the same methodology) leaving German and English vectors of EN-DE unchanged. We run for 100 passes and Vowpal Wabbit fines tunes by itself the learning parameters.

3.3 Trained from Scratch

3.3.1 Cross-Lingual Training using Sentence Alignment (multi(sent_ali))

Europarl Parallel Corpus For sentence alignment, we use a combination of Europarl Parallel Corpus v7.1 (Koehn, 2005), titles from Wikipedia, and parallel news commentary[8]. Extracted from parliament proceedings, Europarl covers over 21 European languages. This extended corpus PC is chosen because it is commonly used in the literature due to its richness and large number of instances. Table 3.1 specifies the train, dev and test distribution for each type of language alignment.

	English-German	English-French	English-Italian
Train	1,751,836	1,858,189	1,613,149
Dev	583,945	619,396	537,716
Test	583,947	619,398	537,718
Total	2,919,728	3,096,983	2,688,583

Table 3.1: Training, Development and Testing split distribution of parallel sentences per language

Bilingual Case Training embeddings by optimizing the cross-lingual objective using sentence alignment means to train a model that maximizes the semantic similarity between parallel sentences. Formally, given pairs of parallel sentences in two languages l_1 and l_2, the goal is to find the embeddings matrices P and Q which transform sentences in l_1 and l_2 to one common space. For that purpose, we minimize the sum of the distances between the embeddings representation of aligned sentences as follows:

$$\begin{aligned} \mathrm{L} = \frac{1}{2 \times N} \times \sum_{i=0}^{N} \parallel P - P_0 \parallel^2 + \frac{1}{N} \times \mu \times \sum_{i=0}^{N} \parallel P^T s_i - Q^T t_i \parallel_1 + \\ \frac{\mu_S}{2} \times \parallel P \parallel_F^2 + \frac{\mu_T}{2} \times \parallel Q \parallel_F^2 \end{aligned} \tag{3.6}$$

[8] This corpus was directly downloaded from http://128.2.220.95/multilingual/data/

where N is the total number of aligned sentences, $s_i \in l_1$, $t_i \in l_2$ and $(s_i, t_i) \in PC$ and μ, μ_S, μ_T are regularization terms. Here l1-distance was chosen instead of l2-distance for its robustness against outliers.

We take advantage of monolingual embeddings to initialize P with P_0. P and Q are optimized using gradient descent with steps $P = step_P \times \delta(P)$ and $Q = step_Q \times \delta(Q)$ to optimize P and Q respectively as follows:

$$\text{step}_P = \frac{\eta}{\epsilon + \sqrt{\| \delta(P) \|^2}} \tag{3.7}$$

$$\text{step}_Q = \frac{\eta}{\epsilon + \sqrt{\| \delta(Q) \|^2}} \tag{3.8}$$

where the gradients are computed as follows:

$$\delta(P) = \frac{\mu}{N} \times S \cdot T \cdot \| P^T s_i - Q^T t_i \|_1 + (\frac{1}{|P|} + \mu_s) \times P - \frac{P_0}{|P|} \tag{3.9}$$

$$\delta(Q) = -\frac{\mu}{N} \times T \cdot T \cdot \| P^T s_i - Q^T t_i \|_1 + \mu_t \times Q \tag{3.10}$$

The list of parameters used for our experiment to generate embeddings is as detailed in table 3.2.

Param	**Val**
μ	1e-9
μ_s	1e-11
μ_t	1e-11
num epochs	50
η	1
ϵ	1e-12
Dimension	300
Learning Rate	10-2
Batch size	64

Table 3.2: Training Parameters for Sentence Alignment

Multilingual Extension The multilingual extension is straightforward as the bilingual objective function is additive. Therefore, the multilingual objective consists of the sum of multiple bilingual objectives which is equivalent to one bilingual objective where the source language for sentences is any non-English language, and the target is English. Thus, we train multilingual embeddings using a concatenation of all sentences from German, French, and Italian to learn P and English sentences to learn Q.

3.3.2 Multilingual Skip-gram ($multi(skip\_gram)$)

We follow the approach [10] described in (Ammar et al., 2016) for extending bilingual Skip-gram defined in (Luong et al., 2015) for multilingual setting. Recall that Skip-gram model as shown in 3.3 is used to learn a probability distribution over words useful for predicting the context of any given word. Bilingual skip-gram extends the monolingual skip-gram by predicting words from both the monolingual and bilingual contexts. This bilingual context for a particular word is built using words neighboring the corresponding aligned word in the parallel sentences. For practical reasons, they assume that sentences are monotonically aligned with words being automatically aligned if they share the same position in the sentence.

[9] Figure taken from Demir and Özgür (2014)

[10] Code and Pre-trained multi-Skip embeddings for up to 59 languages were obtained from http://128.2.220.95/multilingual/data/. We use 512-dimensional embeddings trained on 13 languages for CLDC and CLCD

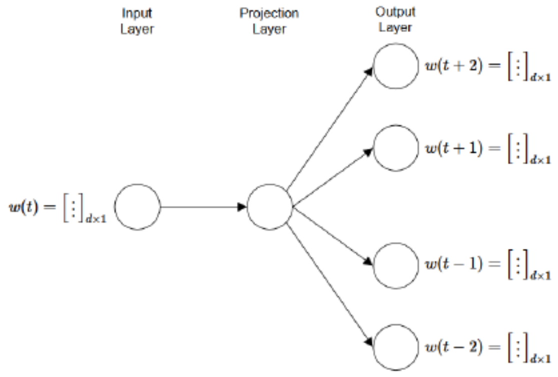
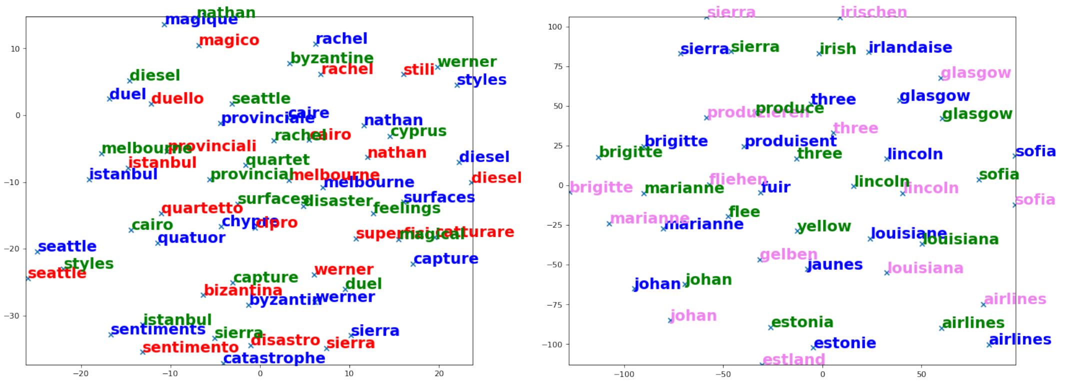

Input Layer
Projection Layer
Output Layer
$w(t) = \left[\vdots\right]_{d\times1}$
$w(t+2) = \left[\vdots\right]_{d\times1}$
$w(t+1) = \left[\vdots\right]_{d\times1}$
$w(t-1) = \left[\vdots\right]_{d\times1}$
$w(t-2) = \left[\vdots\right]_{d\times1}$

Figure 3.3: Skip-gram Model[9]

3.4 Qualitative analysis of Embeddings



(a) English, French and Italian

(b) English, French and German

Figure 3.4: Nearest Neighbours of Cross-Lingual Word Embeddings

Before applying the generated embeddings on downstream tasks, we start by analyzing their quality of coverage and translation. Table 3.3 includes vocabulary coverage for the different embeddings across languages and shows that *pseudo_dict* and *expert_dict* have higher vocabulary coverage while table 3.4 includes coverage against the three downstream tasks. We observe that although *sem* and *sent_ali* have significantly lower vocabulary coverage in general, they tend to cover well the vocabulary of the datasets in our evaluation tasks and this is due to the low occurrence of Out-of-Vocabulary words.

	svd (expert or pseudo)	**CCA**	**sem**	**sent_ali**	**skip_gram**
EN	2,519,369	176,690	76,843	43,741	120,530
DE	2,275,233	376,550	157,192	45,362	137,166
FR	1,152,449	213,578	199,668	40,417	94,251
IT	871,053	233,253	178,965	45,264	90,662
Total	6,818,104	1,000,071	612,668	174,784	442,609

Table 3.3: Vocabulary Coverage for Different Embeddings and Languages

We directly analyze their performance on cross-lingual intrinsic task mainly word translation. For that, we report on the quality of translation matrices for all embeddings. In table 3.5, we report test precision results $P@1$ and $P@5$ which are the percentages of cases where a word in the target language is found in top 1 and 5 of the nearest neighbor respectively of the source word such as the source and target words are paired in the dictionary. We observe that *expert_dict* outperforms other

	svd (expert or pseudo)	**CCA**	**sem**	**sent_ali**	**skip_gram**
CLDC	92.29%	**93.25%**	87.18%	84.22%	88.65%
Churn	**95.43%**	93.73%	91.46%	86.49%	90.2%
Event	-	63.67%	-	-	-

Table 3.4: Vocabulary Coverage for Different Embeddings and Tasks

embeddings for all languages. We also note that although *pseudo_dict* and *expert_dict* have the same vocabulary coverage, they don't have the same quality of translation. CCA has a lower coverage on Event Twitter data because of the presence of misspellings and proper nouns (hyphens in the table mean other embeddings are not tried for this task).

	pseudo		**expert**		**CCA**		**sem**		**sent_ali**		**skip_gram**	
Translation	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5	P@1	P@5
DE->EN	25.1	37.8	**27.9**	**41.8**	26.5	42.9	12.4	29.4	17.5	29.1	26.5	37.3
FR->EN	31.9	43.4	**33.0**	**44.0**	28.3	41.3	5.9	14.8	20.6	29.0	28.9	37.9
IT->EN	41.5	59.2	**45.6**	**62.1**	41.6	59.1	12.6	30.7	27.4	40.0	37.3	51.1

Table 3.5: Quality of Translation Matrices in terms of Precision percentages

Figure 3.4(a) and 3.4(b) give a qualitative visualization of cross-lingual nearest neighbor using *expert_dict* word embeddings aligned in English(green), French(blue), Italian(red) and German(violet). This visualization was realized using TSNE to project 300-dimensional words into 2 dimensions and the words were sampled from the test dictionary where the embeddings achieved a good translation quality. The illustration shows multilingual embeddings are successful to capture the similarity between "magique" and "magico", "duel" and "duello" which shows that learning the alignment from French to English and Italian to English was enough to create a transitive alignment between French and Italian. The same can be said for multilingual word clusters like ("yellow", "gelben" and "jaunes") which are closer to each other and suggest that our methodology could learn a transitive relationship between French and German through English.

3.4.1 Conclusion

In this chapter, we define different multilingual embeddings that are covered in downstream applications in the following sections. Specifically, we include embeddings from different families in our analysis. Some of them are fine-tuned: *pseudo_dict*, *expert_dict* and *sem* while others are trained from scratch: *sent_ali* and *skip_gram*. Some of them are induced from scratch to ensure fair comparison with monolingual embeddings later on. To save time, some of them are obtained directly whenever possible. We also evaluate their performance directly on the tasks of word translation and cross-lingual nearest neighbor and check their vocabulary coverage to get a first-hand idea of their quality. Overall, our preliminary qualitative analysis for fine-tuned methodologies is mostly in favor of *expert_dict*, which reached the best performance in terms of word translation. It remains to test to which extent results for extrinsic evaluation in later chapters are consistent with this intrinsic performance.

4 Cross-Lingual Document Classification

4.1 Overview

Cross-lingual document classification (CLDC) is the first downstream task against which multilingual embeddings induced in the previous chapter are evaluated. Our aim for this task is to investigate the usefulness of a model trained on the aggregation of different languages and to compare its performance to language specific models. Multilingual embeddings are the mechanism used for training across languages by enabling a common representation of documents sharing the same meaning but coming from different languages. We test the ability of that multilingual representation of documents to capture the semantic and syntactic similarities between the languages which help with transfer learning. In other words, we investigate how languages which are resource rich in the current task can help those which lack the features needed to build a strong classifier by their own.

The existing benchmark on Cross-lingual Document Classification uses RCV1/RCV2 corpora and applies a shallow algorithm (averaged perceptron) to train on documents in a source language L_1 and test its direct applicability to documents in a target languages L_2 as depicted in figure 4.1. This pipeline has always been used for the evaluation of bilingual embeddings and their direct transfer from one language to another. In this thesis, our goal is to extend this approach on more than one target language and when both source and target languages are used for training a language agnostic model. We experiment with deeper models for text classification and perform a comparative analysis between different multilingual embeddings and models. Moreover, we explore different techniques for specializing the embeddings to CLDC task by trying joint training and multi-tasking of multilingual embeddings and document classification.

4.2 Datasets

The dataset used in this task is the Reuters RCV1/RCV2 corpora described in Lewis et al. (2004) and obtained by NIST[1]. We choose to work with this dataset since it has been extensively used in prior research on evaluation of multilingual embeddings and is one of the most important and available benchmarks with a sufficient amount of training instances. RCV1 consists of about 810,000 English newswire stories while RCV2 contains over 487,000 news stories in thirteen other languages[2] all made available by Reuters, Ltd. Although this study focuses on four languages spoken in Switzerland: English, French, German and Italian to save time, the described models and evaluation strategy can be extended to more languages. Each newswire document can be labeled with multiple categories at the same time by topic, industry or region. We follow the same cross-lingual document classification benchmark defined in previous work and work on a multi-classification task with at most one single label per document among four topic categories as described in the next section.

[1] http://trec.nist.gov/data/reuters/reuters.html

[2] The thirteen languages are: Dutch, French, German, Chinese, Japanese, Russian, Portuguese, Spanish, Latin American Spanish, Italian, Danish, Norwegian, and Swedish

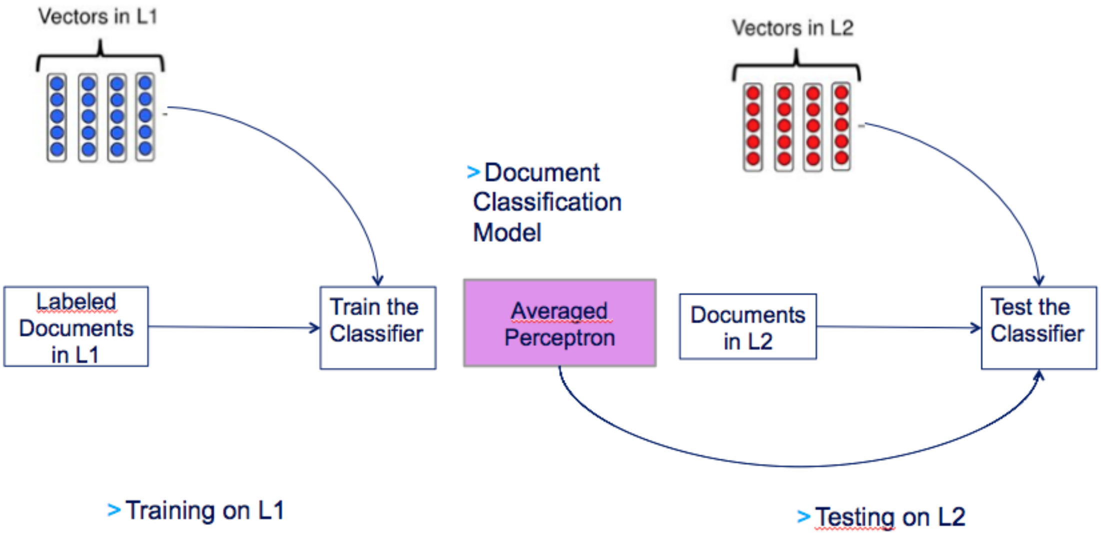


Figure 4.1: Averaged Perceptron for CLDC

	English	**German**	**French**	**Italian**
Train	418,566	50,387	40,470	12,566
Validation	104,601	12,609	10,090	3,129
Test	130,780	15,843	12,669	3,964
Total	653,947	78,839	63229	19,659

Table 4.1: Training, Validation and Testing Distribution of RCV Dataset across Languages

4.2.1 Coarse Grained Dataset

A simpler version of document classification of multilingual RCV corpora[3] was first defined and adopted by (Klementiev et al., 2012). They worked on a coarse-grained single-class classification problem with four classes taken from highest level topics. By noticing that there is a hierarchy of topics and sub-topics, they pick only the higher level topic to be the label of interest when there is only high-level topic among the following: CCAT (Corporate/Industrial), ECAT (Economics), GCAT (Government/Social), and MCAT (Markets). For example, if there is a document with the following multi-class topics: M11, M12, MCAT, then this document is assigned label MCAT (all topics start with the same letter M and M11 and M12 are sub-topics of MCAT). However, if more than one high-level label exists, this document is not considered.

After splitting to 60% train, 20% validation and 20% test, statistics of the dataset are shown in table 4.1. Distribution of labels per language and train/dev/test split is shown in 4.2. Different languages have different dominant classes. The most dominant class in English is CCAT, while GCAT is dominant for German and French and MCAT is for Italian the most prevailing. We choose to keep the dataset imbalanced across languages to synthetically create low-resource vs. high-resource languages across labels and see how languages brought together for training could compensate for that imbalance and the ability of multilingual embeddings in the transfer of annotation between languages. In our case and as table 4.1 shows, English is the dominant language and Italian is the most resource scarce language making up only around 3% of the English dataset.

[3] For copyright limitations, we provide only the ids of train, dev and test documents used in each language in the following repository: https://github.com/meryemmhamdi1/MultilingualThesisDatasets

RCV Dataset Distribution				
Lang	**CCAT**	**ECAT**	**GCAT**	**MCAT**
EN	239,950	43,799	130,442	133,936
DE	17,337	4,396	26,959	15,563
FR	9,322	3,526	33,203	6,685
IT	5,065	3,113	1,473	6,275

Table 4.2: Distribution of Classes in Train/Dev of Coarsed Grained Dataset per languages

4.3 Approach

4.3.1 Data Preprocessing

We perform the following pre-processing operations on our documents:

- *Tokenization:* we use ntlk *sent_tokenize* and *word_tokenize* which are highly efficient and can work for multiple languages. It invokes Treebank tokenizer which splits words based on regular expressions. For the first two models (multi-layer perceptron and convolutional neural networks), we tokenize each document by treating it as one big sentence. While for hierarchical networks and multi-tasking, we keep the hierarchical structure (document -> sentence -> word) where each document is made up of a list of sentences and each sentence is tokenized into a list of words.
- *Stop word and punctuation removal and lowercase conversion:* for each language we remove stop words which are considered as a form of redundant information based on a list obtained from NLTK. We also remove punctuation and convert all words to lowercase to normalize terms, decrease vocabulary size and increase embeddings coverage.
- *Padding with 0s to maximum size:* For models other than multi-layer perceptron, having a fixed size input matrix is a requirement for computing convolutions, encoding with recurrent units and so on. For that reason, we pad to either have fixed length sentences or fixed length documents by adding vectors of 0s
- *Construction of Vocabulary and Conversion of words to ids:* in order to use an embedding layer, all words which are input to the embedding layer need to be converted to numerical ids. Since we are dealing with a multilingual vocabulary, we distinguish between words coming from different languages by adding a language prefix to each word. In this way, two words from different languages but look the same will not be confused. This is a small hack to the problem of false cognates that works well in this case since we don't have many such cases and we know the language of each input document.

4.3.2 Document Classification Models

We survey several neural network models for document classification at different levels of complexity and apply them to the multilingual setting either by directly incorporating different kinds of already trained multi-embeddings or by training the embeddings along with the task. The different variations of the tried models are represented in 4.2. Some models are deeper than others while others are supersets with reduced layers. Our baseline for monolingual document classification is a two-level averaged perceptron applied on top of the average of word features as initially used by Klementiev et al. (2012). In addition to that, we implement and evaluate other extensions namely: Fine-Tuned Perceptron Multi-Filter Convolutional Neural Networks used for feature extraction before applying a dense layer with softmax[4]. Moreover, we explore a model which multi-tasks training multilingual embeddings with document classification. For time limits, we only managed to try multi-tasking multilingual embeddings using sentence alignment along with document classification using HAN-GRU[5].

[4] We use Keras version 2.0.2 for this kind of models

[5] We use Tensorflow version 1.4.0 to implement multi-tasking models as they require lower-level handling of the loss function

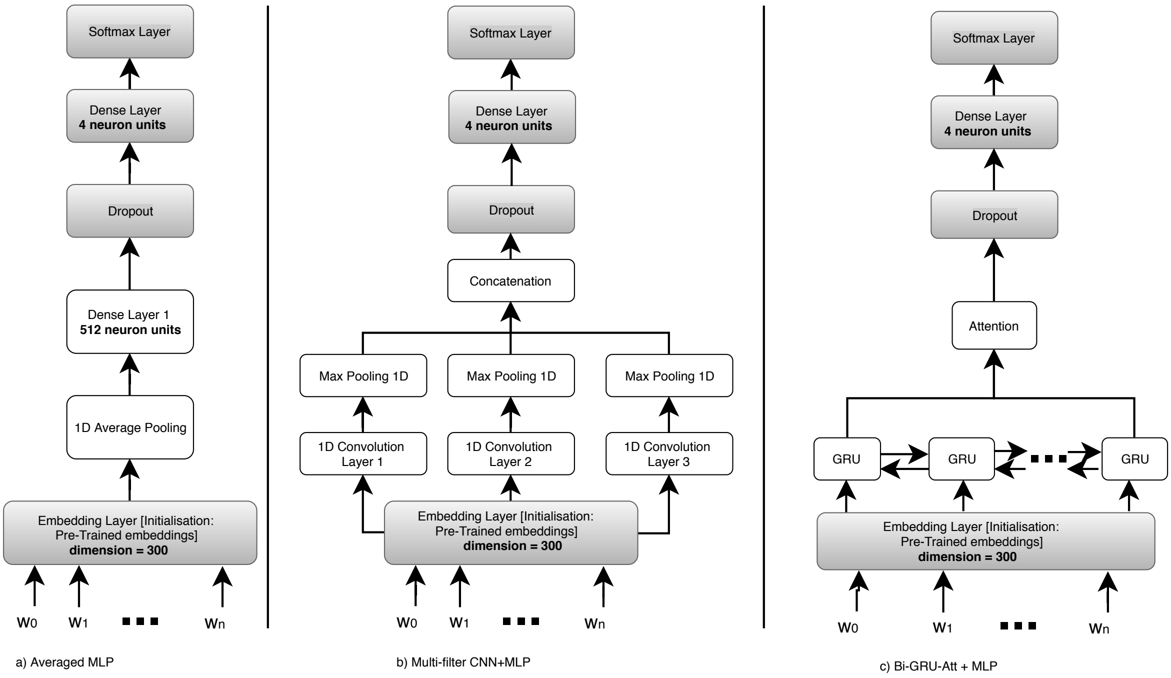


Figure 4.2: Different Document Classification Models

4.3.2.1 Fine Tuned Multilayer Perceptron

A perceptron is a simple linear classifier which predicts whether a feature vector belongs to some specific class by applying a linear predictor function which combines a set of weights with the feature vector as shown in 4.3(a). The mapping from a feature vector to a class prediction is done using the equation:

$$f(x) = \begin{cases} 1 & \text{if } W \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

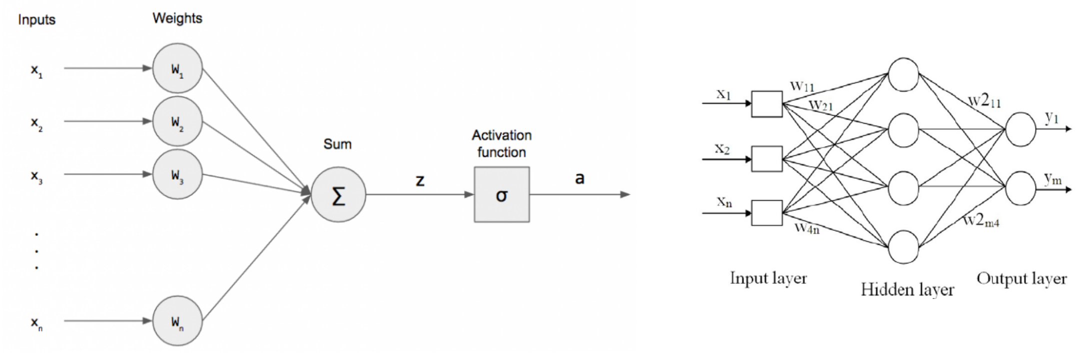
where W is the weight matrix, x the input feature vector and b is the bias. A multilayer perceptron as represented in figure 4.3(b) is an extension of perceptron model where several linear functions are used to classify inputs into several possible outputs, and then the outputs are combined at a later stage. The weights and biases are optimized using backpropagation. We test a shallow version of MLP where only one hidden layer is used. The input to such a network is simply the average of all embedding vectors of the words. To circumvent the padding effect, zero masking is performed so that only non-zero vectors are taken into consideration. The multilingual embeddings are fine-tuned along with other weights in the training process. Working with monolingual and multilingual settings where all embeddings are fine-tuned to the task enables a fair comparison and leads us to conclude on the gain or loss of addition of data from different languages without worrying if the embeddings are specialized enough to the current task.

4.3.2.2 Multi-Filter Convolutional Neural Networks

Motivated by the work of Kim (2014), we build a multi-filter CNN where convolutions with different kernel sizes are applied and concatenated. The architecture is depicted in more details in figure 4.4. Given input text which consists of the concatenation of n words (after padding), an embedding layer is used to convert the words ids into their corresponding m dimensional embedding vectors. The input to the convolution is then the concatenation of the nxm word vectors:

$$\mathrm{x}_{1:n} = x_1 \oplus x_2 ... \oplus x_n \tag{4.2}$$

[6] Diagrams taken from a) https://pythonmachinelearning.pro/perceptrons-the-first-neural-networks/ and b) (Khursiah and Junita, 2013)



(a) Single Perceptron

(b) Multi-Layer Perceptron

Figure 4.3: Single and Multi-layer Perceptron Models[6]

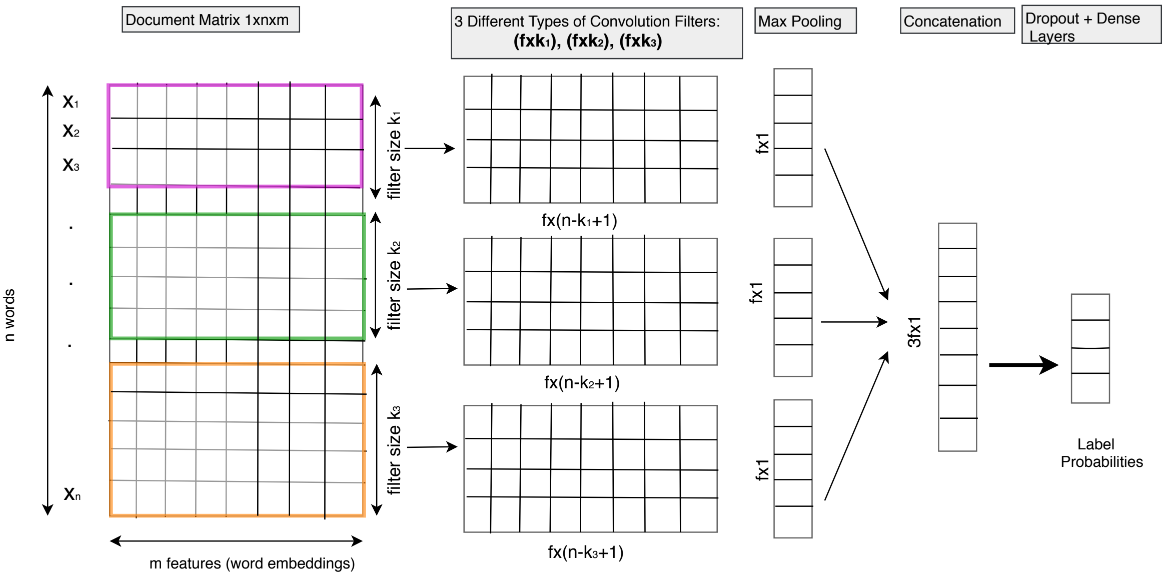


Figure 4.4: Multi-Filter CNN Architecture for Text Classification

We apply a two-dimensional convolution operation which consists of applying a filter of a window of shape: $k \times m$ where k is the number of words and m is the dimensionality (in this case all embeddings dimensions are picked) to be traversed at a time. In other words, a convolution traverses k rows across all m columns of the input matrix at a time, starting from the first row (first word) till the k^{th} word then starts again from the second word till $(k+1)^{th}$ and so on yielding a scalar feature each time. In the end, an output feature o_i is produced from each consecutive window of k words $x_{i:i+k-1}$ using the following equation:

$$\mathrm{c}_i = f(W.x_{i:i+k-1} + b) \tag{4.3}$$

where W and b are the weights and bias terms and f is a non-linearity. The resulting feature vector for each filter of kernel size k is of size $n-k+1$. By applying each filter f times, we obtain $f \times (n-k+1)$ feature maps. In order to concatenate different feature maps from each filter type of sizes (k_1, k_2, k_3 and so on), we apply max pooling first (Collobert et al., 2011). This results in the selection of the maximum feature per each filter. The concatenated feature forms the penultimate layer and a dropout regularization is applied before feeding the output to a dense layer with softmax activation to convert it to a probability distribution over the set of labels.

Loss Function Since we are dealing with a multi-class classification problem where each document can be assigned one label at most, we use weighted categorical cross entropy loss which is defined as follows:

$$\text{L} = \text{-} \textstyle\sum_{i=1}^{n} w_i \times p(i) \times log(\hat{y}_i) \tag{4.4}$$

where n is the number of testing instances, w_i is the weight attributed to each instance corresponding to its class, $p(i)$ is the true label and $\hat{y}_i$ is the prediction. The weights are inversely proportional to the distribution of classes to circumvent the possibility of overfitting that can be caused by imbalanced label distribution as described in table 4.2 and are computed as follows:

$$\text{w}_i = log(\frac{\sum_{i=1}^{n}(max(|y_i|, 1)}{|y_i|}) + 1 \tag{4.5}$$

4.3.2.3 Multi-Tasking Multilingual Embeddings Alignment with Document Classification

Rather than obtaining multilingual embeddings independently and applying them directly to document classification task, we can train them along with the task at hand in an end-to-end multi-tasking fashion. There are two multi-tasking modes: joint and alternate training. Alternate training makes more sense in this current setup since we are dealing with two datasets for the two tasks: parallel sentences for fine-tuning embeddings using sentence alignment and multilingual documents dataset for training document classification models using hierarchical attention network. Figure 4.5 depicts the main components of the followed architecture. The left hand side represents the architecture of the first task that of fine-tuning multilingual embedding using sentence alignment while the right-hand side optimizes for document classification using hierarchical bidirectional GRU attention network. The two tasks share one embedding layer which is tuned by the two tasks. Other layers which are shared between the tasks include word level GRU units and attention activation.

Sentence Alignment On top of the embeddings used to initialize obtained from offline alignment using bilingual dictionaries, this component imposes an additional cross-lingual constraint which relies on sentence alignment. This uses the same principle and dataset as in section 3.3.1. Unlike 3.3.1, we follow a parametric compositional model where each sentence is reduced to a representation that takes into consideration the relationship between words using bidirectional GRU with attention. The goal is to construct sentence embeddings out of word embeddings using the weighted average of the output of bi-GRU states, a representation which can encapsulate word order and importance and is better than taking the plain average of word embeddings.

Let S_i and T_i be the bi-GRU encoded representation of the source and target sentences in the alignment pair (s_i, t_i) respectively. The loss can be defined as reversely proportional to the cosine similarity between each pair (S_i, T_i) in addition to an l2 regularizer added to avoid exploding gradient problem as follows:

$$\text{L} = 1 \text{ - } \frac{\sum_{i=1}^{n} S_i \times T_i}{\sqrt{\sum_{i=1}^{n} S_i^2} \times \sqrt{\sum_{i=1}^{n} T_i^2}} + \frac{1}{2} \times \beta \times \parallel W \parallel_2^2 \tag{4.6}$$

where β is an arbitrarily fixed scalar that is experiment specific and W is the training weights.

Hierarchical GRU-Attention Networks The goal is to use the same bi-GRU architecture employed for learning sentence alignment to jointly learn representation of sentences in the document classification task. This way, our architecture makes use directly of sentence embeddings. Then those sentence embeddings are used to learn document embeddings treating sentences making up the documents the same way words making up sentences were treated. The idea is to capture the hierarchical structure of documents: documents are made by sentences and sentences are made by words. In order words, we construct document representation based on the concatenation of sentence representations where each sentence representation is built from the aggregation of word representations. The intuition behind it is that words and sentences play different roles based on the context in which they occur. The overall architecture is depicted in the right side of figure 4.5 with different levels of representations: word, sentence and document representations and is inspired by the work of (Yang et al., 2016). Hereafter, we describe each component in the hierarchy separately from the lower to the upper level:

- **GRU:** In both word and sentence level representations, bidirectional Gated Recurrent Units (GRU) with attention are used. GRU (Bahdanau et al., 2014) is another compact and lightweight variation of LSTM which drops the forget gate and relies only on reset r_t and update z_t gates. Formally, at time t, GRU computes the output state using the previously hidden state and the update gate as in the equation:

  $$\mathrm{h}_t = (1 - z_t)h_{t-1} + z_t \tag{4.7}$$

  We use GRU instead of LSTM since it gives similar performance while being more computationally efficient as it was shown in the study of (Chung et al., 2014).

- **Sentence Level Representation:** At this level, we are interested in encoding words in each sentence to come up with sentence encoding. Let a sentence s_i consisting of n words: $s_i = [w_{i0}, ...w_{i1}, w_{in}]$ where w_{ij} are the word ids. After converting the word ids to their corresponding embedding vectors $x_{ij} = W_{emb} \times w_{ij}$ where W_{emb} is the embeddings matrix, we use a bidirectional GRU to encode the forward and backward contextual information of the words in each sentence. Those states $fh_{ij} = \overrightarrow{GRU}(x_{ij})$ and $bh_{ij} = \overleftarrow{GRU}(x_{ij})$ computed for each word w_j in sentence s_i where $j \in [1, n]$ are concatenated to form the encoded representation for each word: $h_{ij} = [fh_{ij}, bh_{ij}]$. We apply attention to get a measure of which words are more important by assigning weights of importance. To find those weights, we loop over all encoders' states h_{ij} to compute their scores by feeding them to a dense layer plus non-linearity as in equation 4.8 . Then, those scores are normalized and a probability distribution is obtained using softmax as in 4.9. Then, the sentence representation is the weighted sum of the different encoder states by the attention weights as in 4.10.

  $$\mathrm{u}_{ij} = tanh(MLP(h_{ij})) = tanh(W_w \times h_{ij} + b_w) \tag{4.8}$$

  $$\alpha_{ij} = \frac{exp(u_{ij}^T \times u_w)}{\sum_{j=1}^{n} exp(u_{ij}^T \times u_w)} \tag{4.9}$$

  $$\mathrm{s}_i = \textstyle\sum_{j=1}^{n} \alpha_{ij} \times h_{ij} \tag{4.10}$$

  where W_w and b_w are the weights and bias of the dense layer, u_w is learned during the training process.

- **Document Level Representation:** We obtain document vectors by apply bidirectional GRU with attention on top of the sentence vectors the same way we obtained sentence vectors using word vectors as given by the equations 4.11, 4.12 and 4.13:

  $$\mathrm{u}_i = tanh(MLP(h_i)) = tanh(W_s \times h_i + b_s) \tag{4.11}$$

  $$\alpha_i = \frac{exp(u_i^T \times u_s)}{\sum_{i=1}^{m} exp(u_i^T \times u_s)} \tag{4.12}$$

  $$\mathrm{d}_i = \textstyle\sum_{i=1}^{m} \alpha_i \times h_i \tag{4.13}$$

  where W_s and b_s are the weights and bias of the dense layer, u_s is learned during the training process.

Learning Methodology We alternate between the training of the losses of the two tasks as defined in equation 4.4 and 4.6. Two different optimizers are adapted to each task to make the learning of one task as synchronized as possible with the other one.

4.4 Evaluation Approach

4.4.1 Experiment Design

We evaluate the three text classification models explained in previous section 4.3, where we train several language specific and multilingual models. For each classification architecture, models are trained for each language independently and used as a baseline against one multilingual model:

- **Monolingual EN:** Training on EN using English monolingual embeddings and testing on EN
- **Monolingual DE:** Training on DE using German embeddings and testing on DE
- **Monolingual FR:** Training on FR using German embeddings and testing on FR
- **Monolingual IT:** Training on IT using German embeddings and testing on IT
- **Multilingual:** Training on *All* languages (i.e. $EN + DE + FR + IT$) using multilingual embeddings and testing on EN, DE, FR and IT

4.4.2 Hyperparameters Chart

For each text architecture, we keep the same hyperparameters for both monolingual and multilingual training modes to ensure a fair comparison. Table 4.3 shows the values of the hyperparameters for each model.

Param	**Val**
Dense Units L1	512
Dense Act L1	relu
Dropout	0.7
Optim	Ada
Learning Rate	10-2
Batch size	64

a) FT-MLP

Param	**Val**
Kernel Sizes	3,4,5
# Filters	200
Dropout	0.3
# GRU units	128
Optim	Sgd (10-3)
Batch size	64

b) MF-CNN

Param	**Val**
# GRU units	50
GRU activation	tanh
Dropout	0.5
Optim Task 1	Ada (10-3)
Optim Task 2	Ada (10-2)
beta	1e-10
Batch size	15

c) Multi-Tasking

Table 4.3: HyperParameters for Different Text Classification Architectures

4.4.3 Performance Evaluation Metrics

In order to compare between the performance of the different models, we compute precision, recall, and f1-scores all macro-averaged[7]. It doesn't make sense to compute micro metrics since we are not dealing with a multi-class with multiple labels possible for one instance so we report only macro scores. We define the metrics as follows:

- **Precision score:** is the proportion of correctly predicted instances among the retrieved instances. Given n classes, the macro precision can be computed as:

$$\text{pre} = \frac{1}{N} \sum_{i=1}^{n} P_i \tag{4.14}$$

[7] We follow Scikit-Learn implementation to compute the defined metrics.

such that $P_i = \dfrac{TP_i}{TP_i + FP_i}$ with TP_i and FP_i representing the number of true positive and false positives respectively

- **Recall score**: is the proportion of correctly predicted instances among the total amount of true instances that exists. Given n classes the macro recall can be computed as:

$$\text{rec} = \frac{1}{N}\sum_{i=1}^{n} R_i \tag{4.15}$$

such that $R_i = \dfrac{TP_i}{TP_i + FN_i}$ with TP_i and FN_i representing the number of true positive and false negatives respectively

- **F1-score:** is the harmonic mean of precision and recall scores:

$$\text{f1} = 2 \times \frac{pre \times rec}{pre + rec} \tag{4.16}$$

4.5 Results and Discussion

Tables 4.4 and 4.5 show f1-score, precision and recall of Fined Tuned MLP (FT-MLP) and Multi-Filter CNN (MF-CNN) models trained with different embeddings. These and the following tables in this section and section 5.5 are organized in such a way that results for testing on English, German, French and Italian are reported in the first, second, third and fourth halves respectively (separated by a horizontal line). Each row represents a different training mode of the model where either the test language or the aggregation of all languages is used for training or the embeddings are changed.

In general, for all languages and both text classification architectures, multilingual training wins over monolingual training with an average improvement in F1-score of 4.47% and 2.52% for FT-MLP and MF-CNN respectively. The gain is irrespective of the type of embeddings and text classification architecture used. For FT-MLP, monolingual model performs worst with an average of 82.36% over the four languages, followed by $multi(sent\_ali)$, $multi(sem)$, $multi(CCA)$, $multi(skip\_gram)$, $multi(exp\_dict)$ and $multi(pseudo\_dict)$ with performances of 85.76%, 85.94%, 86.10%, 86.18%, 86.37% and 86.56% respectively. On the other hand, the order of performance for MF-CNN from worst to best is *mono*, $multi(psedo\_dict)$, $multi(sem)$, $multi(exp\_dict)$, $multi(skip\_gram)$ and $multi(sent\_ali)$, $multi(CCA)$ with average F1-scores over all languages of 80.23%, 84.59%, 85.75%, 85.81%, 85.82%, 86.46% and 86.55%. This makes $multi(exp\_dict)$ in the top 2 and $multi(CCA)$ and $multi(skip\_gram)$ in top 3 best models while the order of the other multilingual embeddings depends extensively on the type of the text classification architecture used.

When it comes to the best text classification architecture monolingually, we notice that MF-CNN performs slightly better than FT-MLP with an across language average gain in performance of 2.23% and has a lower gap between multilingual and monolingual, which is not counter-intuitive since MF-CNN is more complex than FT-MLP as it presents more parameters to train which leads even monolingual models to converge better. So, this explains why in the case of FT-MLP, which is more shallow than MF-CNN, we observe a bigger gap of 4.47% between multilingual and monolingual performances. Our working hypothesis is that the more complex a model is the more likely it is able to learn comparatively better by itself while the gain that comes from multilingual aggregation is more pronounced the more shallow the model is.

Our second observation is the lack of a winner among the different embeddings across all languages and for all classification architectures. There is no best model for both setups and for any language. Nevertheless, we can tell that fine-tuned embeddings like $multi(CCA)$ and $multi(pseudo\_dict)$ while not always the best, kept the same consistent strong performance compared to its monolingual counterparts for different languages (except for when tested on English in FT-MLP and tested on German in MF-CNN). Other models like $multi(exp\_dict)$ and $multi(sent\_ali)$ kept the same gain for German, French, Italian across architectures.

Using FT-MLP, the improvement is well pronounced mostly for Italian (the most resource scarce language) with an increase of 7.66% in F1-score achieved with $multi(pseudo\_dict)$ followed by French with an increase of 6.63% achieved with $multi(sent\_ali)$ then German with an increase of 3.2% with $multi(expert_)$. The improvement in f1-score for German, French and Italian using MF-CNN is 2.3%, 3.66% and 2.6% respectively. This finding confirms our hypothesis that the more a language is low resource, the more it is likely to benefit from multilingual training. According to table 4.1, the order of languages in terms of the number of training and validation instances is: Italian, French, German from lowest to biggest which matches the gain order (especially in FT-MLP). Although there is a gain in performance for English, it is marginal for both architectures (only 0.36 at most). Obtaining a monolingual performance for English always on par with multilingual performance is not at all surprising as English is the dominant language accounting for more than 80% of the training and validation data. Therefore, our focus is to show how the multilingual approach can improve other low-resourced languages namely German, French and Italian for which there is always a gain of more than 2% across architectures.

Figure 4.7 shows learning curves using FT-MLP portraying testing performance for each language independently. For French and Italian, we notice that monolingual performance always lags behind all other multilingual models throughout the training process and upon convergence. For German, we notice that different multilingual embeddings reach convergence quicker than monolingual training and that upon convergence, the gap between multilingual and monolingual is still significant. For English, we notice that the performance of both multilingual and monolingual slightly decreases with training. This explains that more training data doesn't help the model as it is already the most dominant language.

Table 4.6 reports on testing results using Multi-Tasking model for learning the embeddings along with the classification task using HAN architecture. We compare its performance to language-independent models using HAN architecture. Due to the complexity of the model and memory issues, we only manage to run for 10K training (and a maximum of 10k for testing in case the number of testing instances is bigger) for a maximum of 4 epochs. This largely explains the lower performance compared to other shallower models like MF-CNN and FT-MLP (for example 91.71% vs. 61.44% for English using FT-MLP vs. HAN respectively). This also enables us to test our hypothesis against a low data regime scenario where no language is predominant and how this impacts the gain in performance of multilingual over monolingual. The results support our assumption by showing a more significant gain of multilingual over monolingual than we were able to observe previously when the whole data was used. The overall gain of multilingual over monolingual amounts to an increase of 10.55% in f1-score. English seems to benefit the most with an increase of 21.4% followed by Italian with an increase of 17.45% and French with an increase of 6.97%. On the other hand, German performs surprisingly well by its own compared to multilingual training generating a loss of 3.6%.

Other multilingual experiments training on English only and testing on other languages have been carried out to test the ability of multilingual embeddings in direct transfer of annotation. However, they are not reported because they do not lead to an overall improvement over monolingual performance for the given dataset and coarse-grained benchmark (e.g. performance on l2 when l1 is used for training is not better than the performance of l1 alone). Our working hypothesis is that the tried models are complex enough to overfit when trained using one specific language and do not generalize well when tested on a different language. So, it is necessary to fine-tune our multilingual models using data from other languages. What we are trying to prove is that the role of multilingual embeddings is to cover the weaknesses of other low resource languages for which there is significantly lower number of instances (non dominant languages namely Italian, French and German) when they are combined in one training set with the dominant language (aka. English). Our main focus in this thesis is to investigate scenarios where multilingual wins over monolingual and find a setup that works well for the multilingual dataset that we possess. Since it is generally hard to obtain large-scale annotated multilingual datasets, there is a shortage of datasets which could be used to test this hypothesis.

4.6 Demo

The demo[8] is a visualization tool which gives an intuitive example of how useful multilingual embeddings can be for potential users. Figure 4.8 shows the welcome page of the demo. To convey the idea, we design and develop two sub-demos which cover two use cases.

[8] The demo can be accessed from https://research.swisscom.ai/multilingual. For credentials, please contact the authors.

Model	Train	Test	Embeddings	F1-Score (%)	Precision (%)	Recall (%)
Fine Tuned MLP (FT-MLP)	**EN**	**EN**	*mono*	91.68	91.62	91.75
	All		*multi(pseudo_ dict)*	91.46	91.21	91.71
			multi(exp_ dict)	91.61	91.40	**91.81**
			multi(CCA)	91.48	91.91	91.05
			multi(sem)	**92.07**	**93.14**	91.02
			multi(sent_ ali)	91.61	91.68	91.55
			multi(skip_ gram)	91.49	91.4	91.58
	DE	**DE**	*mono*	81.65	79.95	83.44
	All		*multi(pseudo_ dict)*	84.44	85.66	83.25
			multi(exp_ dict)	**84.85**	**86.21**	83.54
			multi(CCA)	83.07	82.68	83.46
			multi(sem)	83.15	83.13	83.19
			multi(sent_ ali)	83.93	83.46	**84.42**
			multi(skip_ gram)	83.96	84.35	83.57
	FR	**FR**	mono	81.92	88.44	76.29
	All		*multi(pseudo_ dict)*	88.51	89.54	87.5
			multi(exp_ dict)	88.27	**89.99**	86.62
			multi(CCA)	88.34	88.38	88.31
			multi(sem)	87.75	86.97	88.55
			multi(sent_ ali)	**88.55**	88.55	**88.56**
			multi(skip_ gram)	88.52	88.97	88.07
	IT	**IT**	mono	74.2	77.95	70.8
	All		*multi(pseudo_ dict)*	**81.86**	**84.31**	79.27
			multi(exp_ dict)	80.76	84.15	77.65
			multi(CCA)	81.53	81.17	81.89
			multi(sem)	80.82	82.96	78.80
			multi(sent_ ali)	78.98	82.37	75.87
			multi(skip_ gram)	80.76	81.64	**79.89**

Table 4.4: CLDC Performance Comparison between different training modes with different embeddings using Fine Tuned MLP

4.6.1 Use Case 1: Cross-Lingual Documents in Vector Space

In this option, the user can select a document category or set of categories in RCV coarse-grained dataset (GCAT, MCAT, CCAT, and ECAT) and choose among the list of available languages (English, German, French and Italian). Upon selection, a static 2d plot powered by Plotly.js [9] is displayed with each point representing a document vector. This vector is projected from 512 into 2 dimensions using TSNE. The high dimensional vectors are already precomputed using the activation of the hidden layer before the dense softmax layer in Fine-Tuned MLP Model with $multi(skip\_gram)$ embeddings applied to the test split dataset of CLDC. To remove document cluttering and improve the visibility of the plot, we keep only document points that have more than k neighbors (in this case 25) within a radius of 2.5. The goal is to show that documents belonging to the same category irrespective of the language are closer in space than documents from other categories.

Figure 4.9 shows the output upon selecting all categories and languages. Different languages are represented by different colors and different shapes represent distinct document categories. It is clear from the figure that documents having the same category are closer to each other. In precise terms, documents of class MCAT are all clustered at the bottom left of the figure, while documents with class GCAT in the top right and ECAT in the middle right whereas CCAT at the middle left. Figure 4.10 shows an example when only two languages are chosen for better visibility. Beside our observation in the first example, it is worth noting our multilingual embeddings played an important role in placing documents talking about Economics (ECAT) and those talking about Markets (MCAT) closer to each other than are Government/Social (GCAT) documents to MCAT.

[9] https://plot.ly/javascript/

Model	Train	Test	Embeddings	F1-Score (%)	Precision (%)	Recall (%)
Multi-Filter CNN (MF-CNN)	EN	EN	*mono*	90.34	91.24	89.59
	All		*multi(pseudo_ dict)*	90.46	90.42	90.52
			multi(exp_ dict)	90.34	90.45	90.26
			multi(CCA)	**90.8**	**90.88**	90.72
			multi(sem)	90.14	90.85	89.48
			multi(sent_ ali)	90.18	90.13	90.24
			multi(skip_ gram)	**90.66**	90.69	**90.64**
	DE	DE	*mono*	84.11	85.88	82.42
	All		*multi(pseudo_ dict)*	83.77	84.64	82.97
			multi(exp_ dict)	86.37	83.91	**88.97**
			multi(CCA)	84.46	85.5	83.45
			multi(sem)	83.79	84.13	83.46
			multi(sent_ ali)	**86.41**	**89.76**	83.31
			multi(skip_ gram)	84.52	85.21	83.84
	FR	FR	mono	85.77	88.55	83.17
	All		*multi(pseudo_ dict)*	88.69	88.72	88.66
			multi(exp_ dict)	88.03	88.83	87.25
			multi(CCA)	**89.47**	**90.76**	88.21
			multi(sem)	88.55	88.85	88.26
			multi(sent_ ali)	89.43	90.11	**88.75**
			multi(skip_ gram)	89.16	90.04	88.29
	IT	IT	mono	78.16	81.06	75.47
	All		*multi(pseudo_ dict)*	80.11	83.41	77.07
			multi(exp_ dict)	78.56	81.40	75.92
			multi(CCA)	**81.78**	**84.67**	79.09
			multi(sem)	80.76	81.81	**79.74**
			multi(sent_ ali)	80.18	84.14	76.57
			multi(skip_ gram)	81.49	84.07	79.07

Table 4.5: CLDC Performance Comparison between different training modes with different embeddings using Multi-Filter CNN

4.6.2 Use Case 2: Cross-Lingual Nearest Neighbours of Documents

In the second case, the goal is to choose a document in any language and to find its cross-lingual nearest neighbors. The document can be either selected from the samples or a Wikipedia article whose URL can be entered by the user. Unlike the previous sub-demo, this one is not static as the output depends on the entered document or URL. This requires a call to the backend which runs preprocessing on the fly on the document and use multilingual embeddings *multi*(*skip_gram*) to predict the class of the document using Fine-Tuned MLP and then finds its cross-lingual neighbor among already computed 2D representation of multilingual documents (belonging to Reuters Corpus) in the vector space. Figure 4.11 shows the input to the demo which consists of Reuters documents in four languages and a textbox to enter URL to Wikipedia article. Figure 4.12 shows the nearest neighbors output for the chosen document represented by the big green point in the middle. We notice that there is significantly more nearest neighbors in the same class as the document and they are from different languages. We provide some more examples in the appendix A.

4.7 Conclusion

In this chapter, we design experiments for evaluating multilingual embeddings for the task of Cross-lingual Document Classification using text classification models at different levels of complexity. We not only investigate the performance of embeddings trained independently but we also involve some of our models in jointly fine-tuning with the current task. Due to the similarity between the tasks of training embeddings (using sentence alignment) and document classification, we explore the possibility of multi-tasking both tasks in one architecture attempting to offer an end-to-end solution setup. In general, we notice a gain in the performance of multilingual over monolingual training.

Model	**Train**	**Test**	**Embeddings**	**F1-Score (%)**	**Precision (%)**	**Recall (%)**
Multi-Tasking (HAN+Sent-Ali)	**EN**	**EN**	*mono*	61.44	53.47	72.22
	All		*multi*	**82.84**	**85.62**	**80.27**
	DE	**DE**	*mono*	**89.53**	**86.66**	**92.59**
	All		*multi*	85.93	86.11	85.83
	FR	**FR**	mono	76.89	82.22	72.22
	All		multi	**83.86**	**83.05**	**84.72**
	IT	**IT**	mono	57.27	53.47	61.66
	All		multi	**74.72**	**73.61**	**76.11**

Table 4.6: CLDC Performance Comparison between different training modes with different embeddings using Multi-Tasking HAN and Sentence Alignment

Multilingual embeddings and training are winners the less complex the model and for lower and imbalanced data regimes. We conclude that the gain is more pronounced in the case of low-resourced languages namely French and Italian where multilingual embeddings act like an important agent of transfer of annotation. Multi-tasking with low data regime performed the best in terms of the gap between multilingual and monolingual performance. However, due to memory limitations, we are not able to conclude whether the gain comes exactly from the embeddings learned with multi-tasking or from the low data regime setup. In the end, we make available a visualization tool that shows cross-lingual nearest neighbor applied to the task of document classification on our dataset.

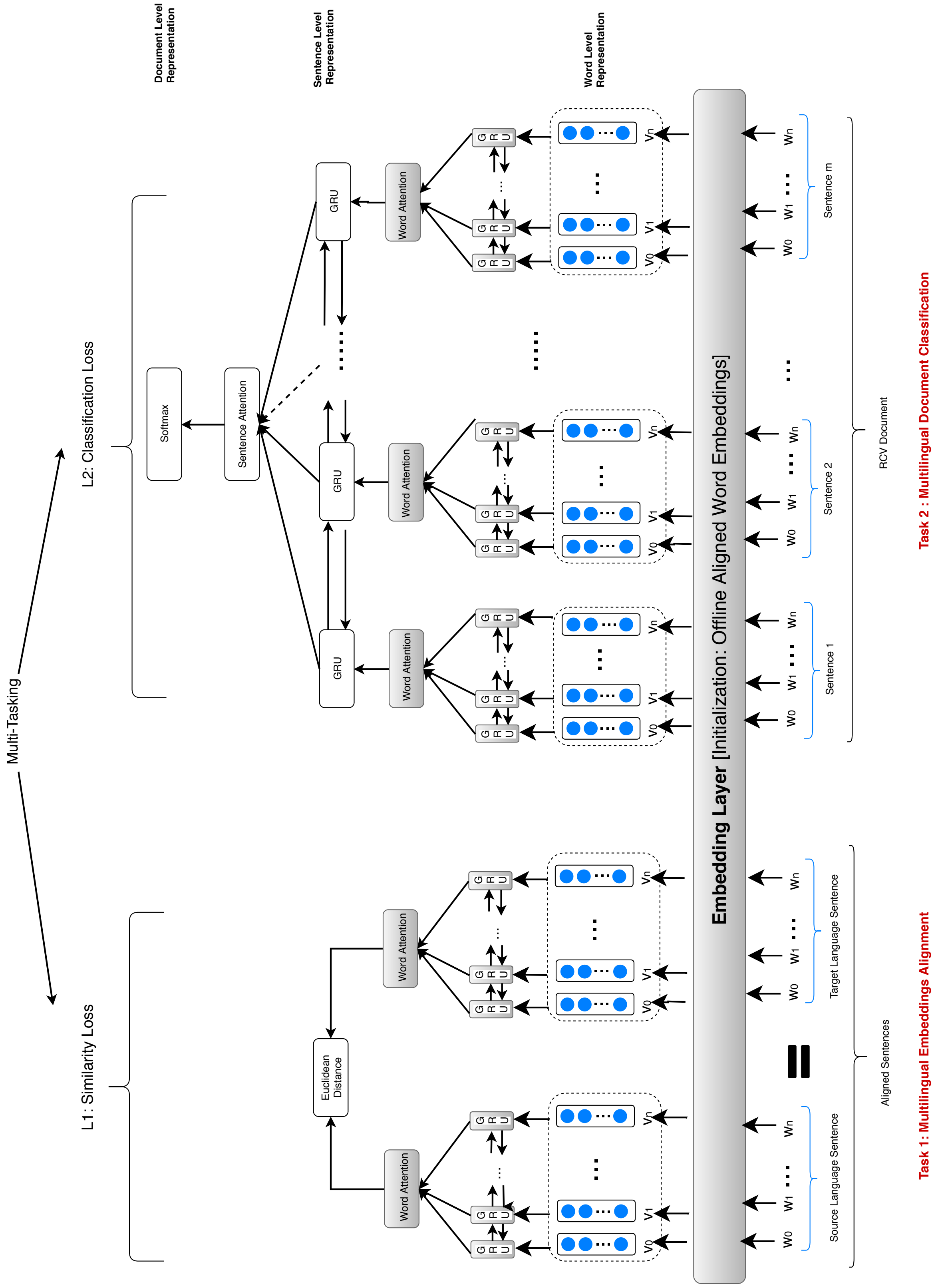
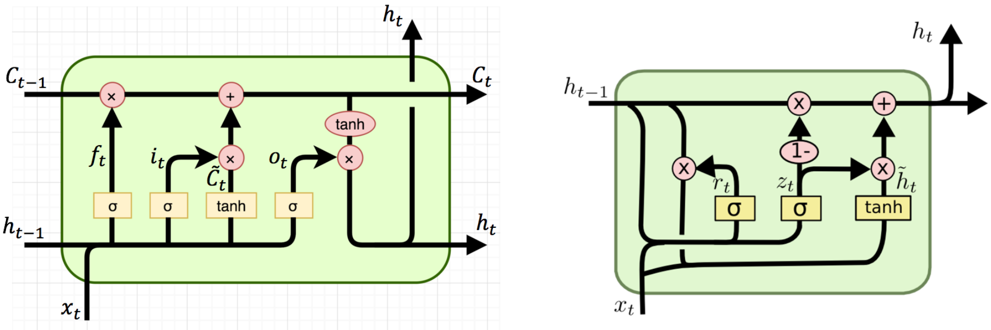


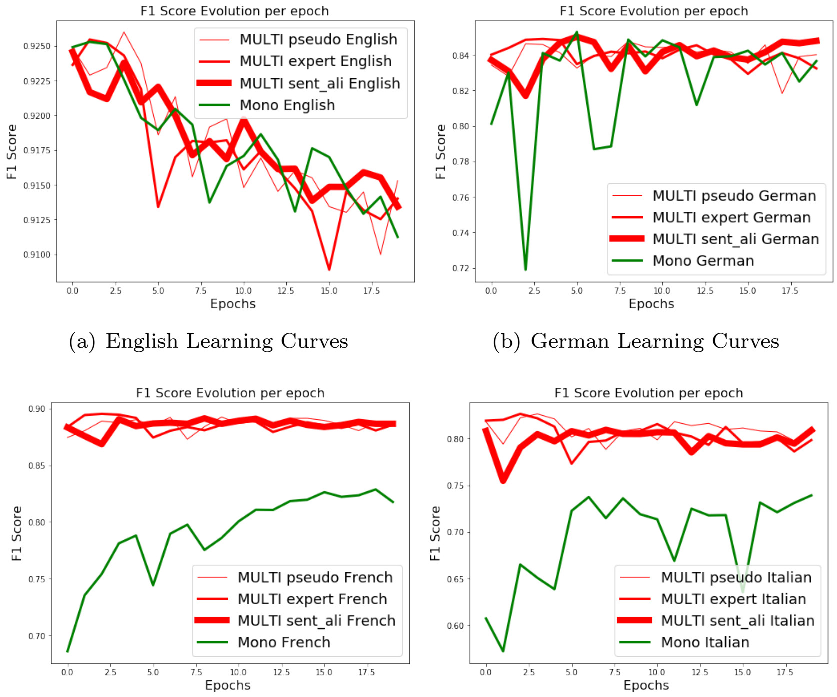
Figure 4.5: Multi-Tasking Hierarchical Attention Networks for Document Classification and Multi-lingual Embeddings Alignment



(a) Long Short-Term Memory

(b) Gated Recurrent Unit

Figure 4.6: LSTM vs. GRU



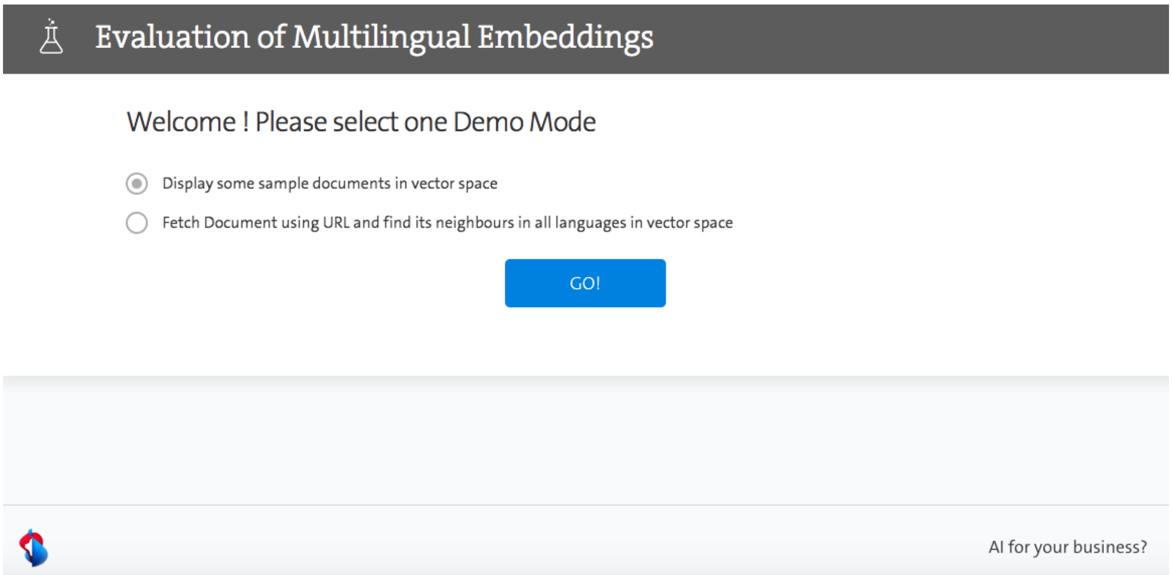
(a) English Learning Curves

(b) German Learning Curves

(c) French Learning Curves

(d) Italian Learning Curves

Figure 4.7: Learning Curves for Fine Grained RCV: multilingual vs. monolingual embeddings



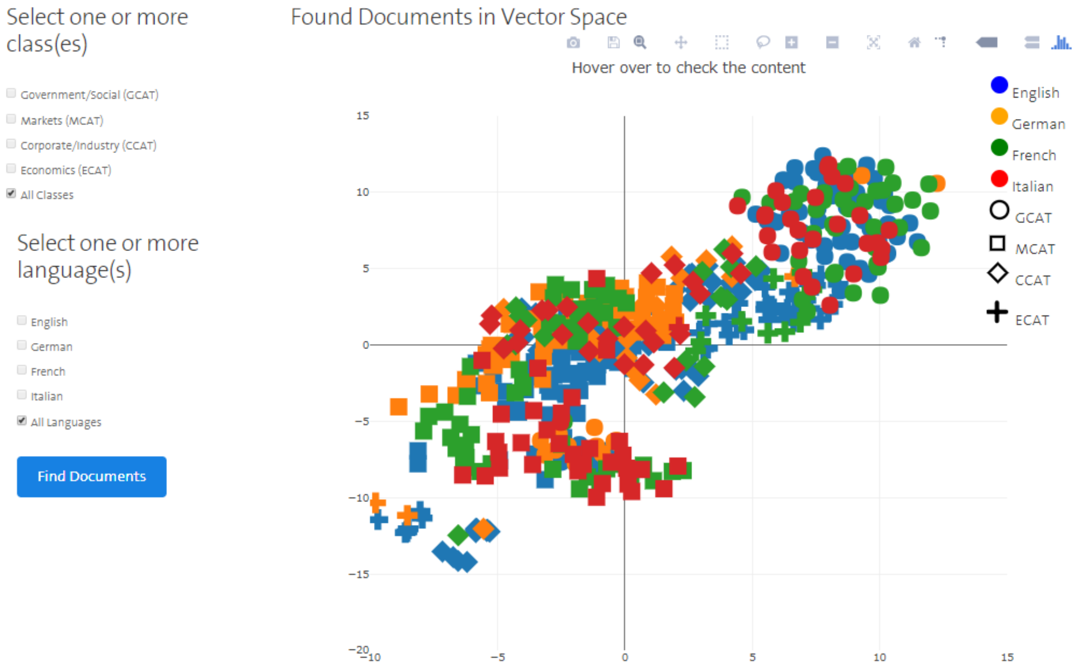
Evaluation of Multilingual Embeddings

Welcome ! Please select one Demo Mode

● Display some sample documents in vector space

○ Fetch Document using URL and find its neighbours in all languages in vector space

Figure 4.8: Demo: Welcome Page



Select one or more class(es)

□ Government/Social (GCAT)

□ Markets (MCAT)

□ Corporate/Industry (CCAT)

□ Economics (ECAT)

☑ All Classes

Select one or more language(s)

□ English

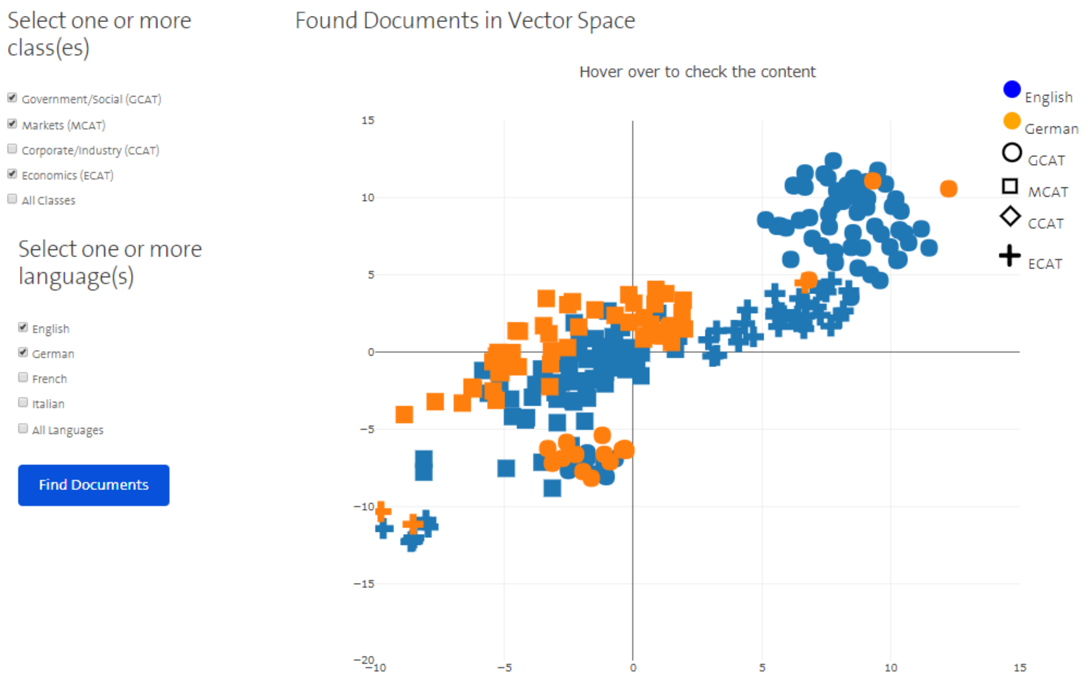
□ German

□ French

□ Italian

☑ All Languages

Figure 4.9: Demo: Option 1 Example 1



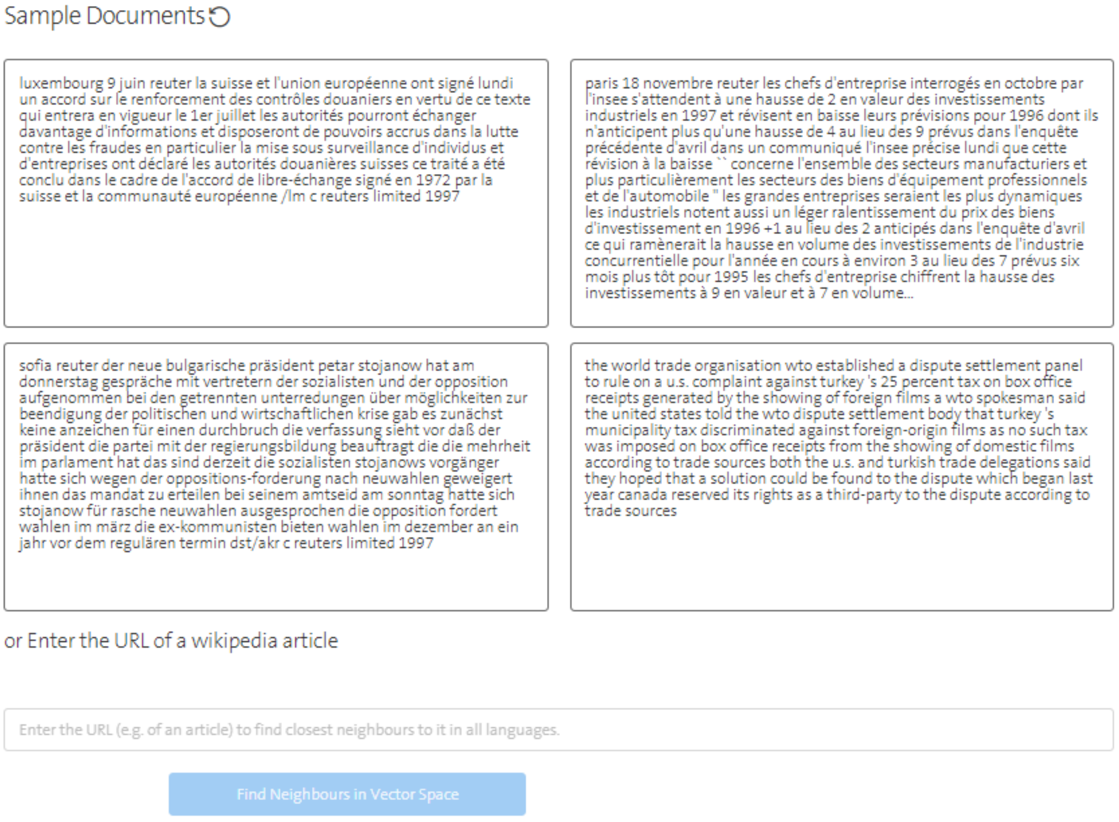
Select one or more class(es)

☑ Government/Social (GCAT)
☑ Markets (MCAT)
☐ Corporate/Industry (CCAT)
☑ Economics (ECAT)
☐ All Classes

Select one or more language(s)

☑ English
☑ German
☐ French
☐ Italian
☐ All Languages

Figure 4.10: Demo: Option 1 Example 2



luxembourg 9 juin reuter la suisse et l'union européenne ont signé lundi un accord sur le renforcement des contrôles douaniers en vertu de ce texte qui entrera en vigueur le 1er juillet les autorités pourront échanger davantage d'informations et disposeront de pouvoirs accrus dans la lutte contre les fraudes en particulier la mise sous surveillance d'individus et d'entreprises ont déclaré les autorités douanières suisses ce traité a été conclu dans le cadre de l'accord de libre-échange signé en 1972 par la suisse et la communauté européenne /lm c reuters limited 1997

paris 18 novembre reuter les chefs d'entreprise interrogés en octobre par l'insee s'attendent à une hausse de 2 en valeur des investissements industriels en 1997 et révisent en baisse leurs prévisions pour 1996 dont ils n'anticipent plus qu'une hausse de 4 au lieu des 9 prévus dans l'enquête précédente d'avril dans un communiqué l'insee précise lundi que cette révision à la baisse `` concerne l'ensemble des secteurs manufacturiers et plus particulièrement les secteurs des biens d'équipement professionnels et de l'automobile " les grandes entreprises seraient les plus dynamiques les industriels notent aussi un léger ralentissement du prix des biens d'investissement en 1996 +1 au lieu des 2 anticipés dans l'enquête d'avril ce qui ramènerait la hausse en volume des investissements de l'industrie concurrentielle pour l'année en cours à environ 3 au lieu des 7 prévus six mois plus tôt pour 1995 les chefs d'entreprise chiffrent la hausse des investissements à 9 en valeur et à 7 en volume...

sofia reuter der neue bulgarische präsident petar stojanow hat am donnerstag gespräche mit vertretern der sozialisten und der opposition aufgenommen bei den getrennten unterredungen über möglichkeiten zur beendigung der politischen und wirtschaftlichen krise gab es zunächst keine anzeichen für einen durchbruch die verfassung sieht vor daß der präsident die partei mit der regierungsbildung beauftragt die die mehrheit im parlament hat das sind derzeit die sozialisten stojanows vorgänger hatte sich wegen der oppositions-forderung nach neuwahlen geweigert ihnen das mandat zu erteilen bei seinem amtseid am sonntag hatte sich stojanow für rasche neuwahlen ausgesprochen die opposition fordert wahlen im märz die ex-kommunisten bieten wahlen im dezember an ein jahr vor dem regulären termin dst/akr c reuters limited 1997

the world trade organisation wto established a dispute settlement panel to rule on a u.s. complaint against turkey 's 25 percent tax on box office receipts generated by the showing of foreign films a wto spokesman said the united states told the wto dispute settlement body that turkey 's municipality tax discriminated against foreign-origin films as no such tax was imposed on box office receipts from the showing of domestic films according to trade sources both the u.s. and turkish trade delegations said they hoped that a solution could be found to the dispute which began last year canada reserved its rights as a third-party to the dispute according to trade sources

or Enter the URL of a wikipedia article
Enter the URL (e.g. of an article) to find closest neighbours to it in all languages.
Find Neighbours in Vector Space

Figure 4.11: Demo: Option 2 User Input

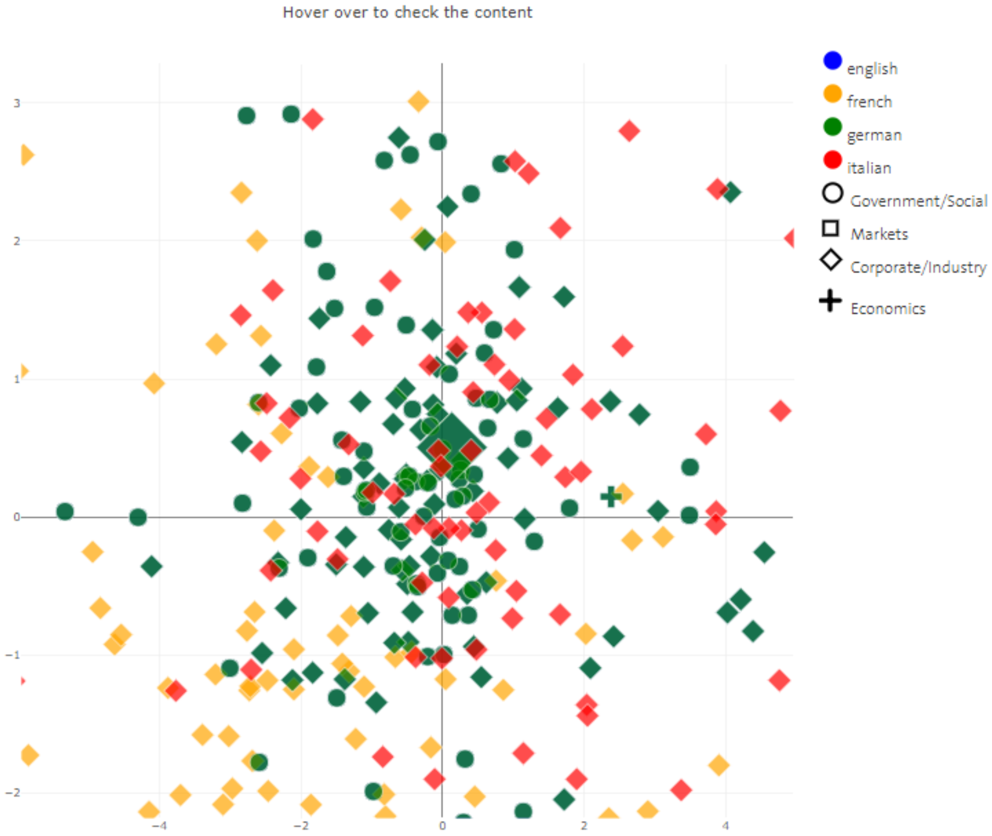


Figure 4.12: Demo: Option 2 Output

5

Cross-Lingual Churn Detection in Social Media and Chatbot Conversations

5.1 Overview

In evaluating our multilingual embeddings, we investigate with other variations of text classification namely churn detection in social media and churn intent detection in chatbot conversations. For those closely related tasks, we investigate the usefulness of a multilingual model that can make use of several languages at the same time to build a strong model that overcomes data sparsity that language specific models suffer from. Since no previous work adopts a multilingual approach for this problem, we get our inspiration from CLDC and build a similar approach by focusing on two languages. Our aim with the multilingual approach is to bridge the gap between English and German and improve the performance of German for which churn detection data lacks what it takes to learn a robust model as we will show in the following sections. Moreover, we investigate the performance of multilingual models trained on social media and their ability to generalize when directly applied to chatbot conversation data. Figure 5.1 shows an overall view of the pipeline.

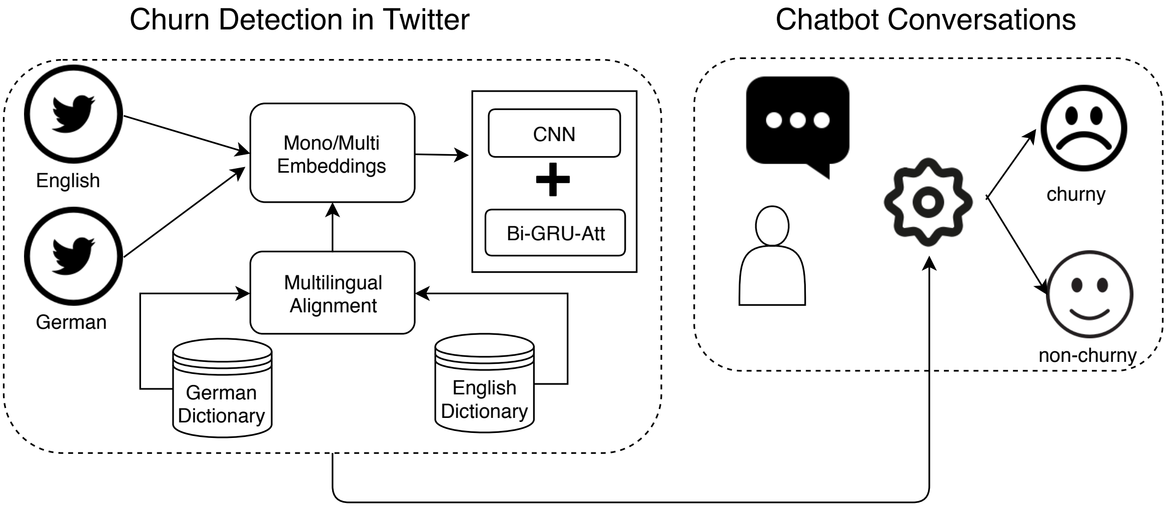


Figure 5.1: Multilingual Churn Detection Pipeline

5.2 Datasets

5.2.1 Churn Detection in Social Media

We use datasets from two languages: English and German. English dataset, which we refer to in the rest of this report as EN_T, is obtained from Amiri and Daume (2015) and contains tweets mentioning one of the following telecommunication brands: Verizon, AT&T and T-Mobile. In other

Twitter English Data (EN_T)		
brand	**churn**	**non churn**
Verizon	447	1543
AT&T	402	1389
T-Mobile	95	978

Table 5.1: Distribution of English tweets along the different brands.

Twitter German Data (DE_T)		
brand	**churn**	**non churn**
O2	247	905
Vodafone	203	1061
Telekom	121	1397
Others	40	365

Table 5.2: Distribution of German tweets along the different brands.

Chatbot Conversation Data		
Lang	**churn**	**non churn**
EN	119	188
DE	116	218

Table 5.3: Distribution of labels in chatbot conversations for both languages (EN/DE).

words, a churny tweet that mentions a particular source brand means that the Twitter user expresses an intent to leave that brand. There are 4339 tweets in total with annotation confidence above 0.7. The distribution of the dataset with respect to the source brands is shown in 5.1.

German dataset, later referred to as DE_T, was obtained from joint work with Christian Abbet on a shared publication with equal contribution (Abbet et al., 2018). They crawled and applied filters based on keywords specifically predefined to detect potential churny tweets and only this filtered subset is manually annotated. Then, tweets labeled with high confidence are used to bootstrap another subset of the initially crawled that was not filtered by the special keywords. The distribution of churny tweets vs. non-churny with respect to the source brands is shown in 5.2.

5.2.2 Churn Intent Detection in Bot Conversations

The same study that generated German churn dataset is the one from which chatbot conversations are obtained. They use a user-friendly platform for crowdsourcing this kind of data using chatbot interface that simulates to a great extent customer service. The same model trained on churn detection in social media is what helps the users annotate sentences they are asked to enter as either churny or non churny. Then, their approval of the correctness of the label is asked as the last step to help the model learns from its mistakes. The distribution of churn in the resulting chatbot conversations is tabulated in 5.3. To simplify notation, we use EN_B to refer to English conversational bot, DE_B for German conversational bot and $(EN + DE)_B$ as the concatenation of both English and German bot conversations.

5.3 Models

Like CLDC, we survey several text classification models[1] for this task against different embeddings models. In addition to Fine-Tuned MLP and Multi-Filter CNN explained in previous chapter 4, we experiment with some state-of-the-art models for short text classification namely Bidirectional GRU with Attention and a combination of CNN and GRU with attention. We could not explore those models earlier with document classification as they are expensive to train given the large number of maximum tokens per document. We instead tried hierarchical networks with multitasking which doesn't treat each document as a long sentence but rather splits it into several sentences which capture

[1] We use Keras 2.0.2 as our Neural Network API as only standard models for text classification are implemented for churn detection

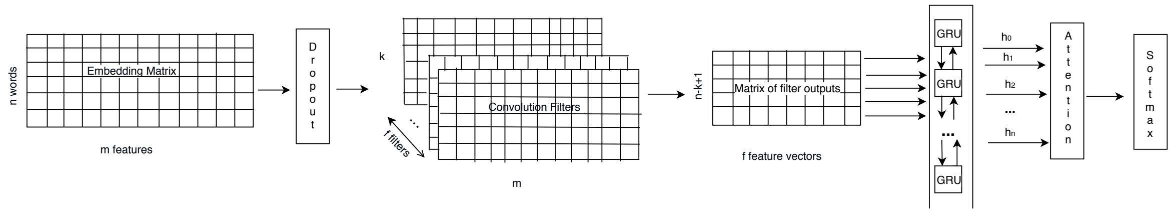


Figure 5.2: CNN-biGRU-Att Architecture

the hierarchical nature of documents. Since we have already explained FT-MLP and MF-CNN models in the previous chapter 4, we dedicate this section to define the two left models.

5.3.1 Bidirectional GRU with attention (bi-GRU-Att)

We use a non-hierarchical version of Bidirectional GRU with attention model described in section 4.3.2.3. We encode each tweet using GRU in the forward and backward directions and apply word level attention as described by 4.8, 4.9, and 4.10 to come up with tweet level representation.

5.3.2 Cascaded CNN and Bidirectional GRU with attention (CNN-biGRU-Att)

We build CNN-biGRU-Att, a cascaded architecture where CNN and biGRU with attention are employed one after the other to complement each other. Rather than relying on CNN or BiGRU independently, it has been proved that combining them in one architecture can be beneficial. This makes it possible to build a classifier that takes advantage of the best of the two worlds. If CNN are known to simulate the role of n-grams feature extraction, GRU take context and word order into consideration. Like in CLDC, we only experiment with BiGRU instead of BiLSTM for a comparable performance with significantly better computational efficiency.

Figure 5.2 shows an overview of the overall architecture and the layers involved. For the sake of simplicity, the diagram is for one tweet at a time. Given an embedding matrix which represents the features of one tweet which consists of n words after padding to the maximum number of words found in the longest tweet and where m is the number of features or dimensionality of word embeddings. We apply dropout directly to the input embedding matrix as it is shown to reduce overfitting the earlier it is used. Then, for each tweet matrix, we obtain f vectors of size n-k+1 by applying convolution filters of kernel size k each. For simplicity, we apply one type of filter as a multi-filter CNN would require an additional level of aggregation. The extracted features are fed to a bidirectional GRU which traverses them both in the forward and backward directions. Attention is used on top of that to come up with a weighted sum of the features before using a softmax activation function to get the final prediction.

5.4 Evaluation Methodology

5.4.1 Experiment Design

We design different experiments for the evaluation of churn detection in social media and conversation bots where we train several language specific and multilingual models using different text classification architectures. For churn detection in social media, we evaluate four models: FT-MLP, MF-CNN, bi-GRU-Att, and CNN-biGRU-Att. For bot conversation, we directly apply the models trained on social media dataset. In both cases, models are trained for each language independently and used as a baseline against multilingual models:

- Training on EN_T using English embeddings and testing on EN_T
- Training on DE_T using German embeddings and testing on DE_T
- Training on $(EN + DE)_T$ using multilingual embeddings and testing on EN_T and DE_T
- Training on EN_T using English embeddings and testing on EN_B
- Training on DE_T using German embeddings and testing on DE_B

- Training on $(EN + DE)_T$ using multilingual embeddings and testing on EN_B and DE_B

5.4.2 Hyperparameters and Word Embeddings

Param	Val
Dense Units L1	512
Dense Act L1	relu
Dropout	0.7
Optim	Ada
Learning Rate	10-2
Patience	20

a) FT-MLP

Param	Val
Kernel Sizes	3,4,5
# Filters	200
Dropout	0.3
Optim	Ada
Learning Rate	10-3
Patience	20

b) MF-CNN

Param	Val
# GRU units	150
GRU activation	tanh
Dropout	0.3
Optim	Ada
Learning Rate	10-3
Patience	20

c) bi-GRU-Att

Param	Val
Kernel Size	2
# Filters	256
# GRU units	128
Dropout	0.3
Optim	Ada
Learning Rate	10-3
Patience	20

d) CNN-biGRU-Att

Table 5.4: HyperParameters for Different Text Classification Architectures

For all experiments, consistent training conditions are adopted in order to ensure a fair comparison between monolingual and multilingual settings. In other words, the same hyper-parameters are used in the design of each text classification architecture independently. Those parameters differ only between architectures but not within different training modes (monolingual vs. multilingual) of the same architecture. Table 5.4 shows the different hyperparameters used for each model. For FT-MLP, we use a first dense layer with 512 units and rectified linear unit activation prior to the second dense layer that directly precedes softmax activation, a dropout layer of 0.7, an Adam optimizer with learning rate 10-2. For MF-CNN, we use 3 types of filters with kernel sizes 3, 4 and 5 consisting of 200 filters each, a dropout of 0.3 and Adam optimizer with learning rate 10-3. Bi-GRU-Att uses 150 GRU units with tanh as an activation function, dropout layer of 0.3 and Adam optimizer 10-3. For CNN-biGRU-Att, we employ 256 convolution filters with a kernel size of 2, 128 GRU units and apply a dropout with rate 0.3. In all experiments, we train for 20 epochs and use early stopping with patience 20 [2].

Monolingual Word embeddings are obtained directly from FastText (Bojanowski et al., 2017), while there are different flavors of multilingual embeddings evaluated for all architectures (except CNN-biGRU-Att where only the best model $multi(exp\_dict)$ is used). For models trained using CNN-biGRU-Att and tested on social media datasets, we use a 10-fold cross-validation to ensure that the train/test split doesn't affect much the results. For models tested on chatbot conversations, cross-validation doesn't apply since we only take the best model trained on social media datasets and directly apply it. In the end, we report on F1-scores, precision and macro scores or their means and standard deviation if cross-validation is used.

5.5 Results and Discussion

5.5.1 Social Media Churn Results

Table 5.5 contains results obtained for churn detection in social media using FT-MLP across different training modes and embeddings. The results show that in general multilingual models always outperform monolingual baselines for both English and German irrespective of the embeddings model used with an overall increase of 14.08%. As a matter of fact, the best model for English EN_t is obtained by training on $(EN + DE)_t$ using multilingual embeddings obtained using SVD with pseudo dictionary $multi(pseudo\_dict)$ with an F1-score of 71.84% outperforming monolingual by a margin of 3.8%. In the second and third position come multilingual embeddings obtained with $multi(CCA)$ and multilingual embeddings trained jointly $multi(skip\_gram)$, whereas monolingual only beats multilingual embeddings obtained with semantic specialization $multi(sem)$ and only for English. Similarly, the best model for German DE_t is the one trained on $(EN + DE)_t$ using multilingual embeddings

[2] Keras version 2.0.2 implementation and code executed on Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz

$multi(sent\_ali)$ while $multi(CCA)$ and $multi(sem)$ come in the second and third places and monolingual is the last on the list way behind the other models with an underperformance of 10.22% over the best model. Against our expectations, $multi(exp\_dict)$ lagged behind other models in this case. This difference between the degree of improvement of multilingual vs. monolingual for English and German is due to the fact that English dataset has already what it takes to learn classification patterns while German benefits more from the aggregation of more languages to learn more complex patterns that are not present in German alone. So, for this model, multilingual training is a clear winner where the order of performance from best to worst is $multi(pseudo\_dict)$, $multi(cca)$, $multi(sent\_ali)$, $multi(skip\_gram)$, $multi(sem)$, $multi(exp\_dict)$ then $mono$ at the end with f1-score averages over EN and DE of 67.96%, 67.95%, 67.43% , 67.06%, 66.13%, 66.07% and 61.75%. We notice that multilingual embeddings performance are close to each other with an average of 67.1 ± 0.86 and far from monolingual performance.

The results for MF-CNN are depicted in table 5.6. Unlike FT MLP, the best performant model for En_T is the one trained with $multi(exp\_dict)$ with an F1 score of 79.43% which is 2.41% more than monolingual model whereas the second place goes to the model trained with $multi(pseudo\_dict)$ with a close performance of 79.31%. Monolingual model performance is comparable to them with a performance of 77.02%. Then come other multilingual models namely $multi(skip\_gram)$, $multi(CCA)$ and $multi(sem)$ and $multi(sent\_ali)$ which couldn't outperform monolingual model. The same thing cannot be said for DE_t for which all multilingual models outperform the monolingual model with the best model achieved using $multi(pseudo\_dict)$ reaching an F1-score of 69.03% which accounts for a significant increase of 10.45% over monolingual performance. In this case, multilingual training is still the winner for both languages with an overall increase of 12.86%. The order of performance from best to worst is $multi(pseudo\_dict)$, $multi(exp\_dict)$, $multi(skip\_gram)$, $multi(sem)$, $multi(cca)$, $mono$ then $multi(sent\_ali)$ at the end with f1-score averages over EN_t and DE_t of 74.17%, 73.12%, 72.73% , 70.91%, 70.60%, 68.0% and 67.51%.

When applying biGRU-Att for this problem, we notice that the gap between multilingual and monolingual is smaller as the performance is better for German with an increase of 11.29% achieved with $multi(pseudo\_dict)$ while there is a decrease of 2.85% for English which means that there is an overall improvement of 8.44% for both languages compared to 12.86% in case of MF-CNN and 14.08% with FT-MLP. This emphasizes our previous observation in CLDC that the more sophisticated the model is, the better is the monolingual training by itself. The order of performance of the different models from best to worst is as follows: $multi(exp\_dict)$, $multi(pseudo\_dict)$, $multi(cca)$, $multi(sem)$, $mono$, $multi(sent\_ali)$ and $multi(skip\_gram)$ with average cross-lingual F1-scores of 73.49%, 73.36%, 71.79%, 69.89%, 69.14%, 68.28% and 63.53%.

Although there is no clear winner in terms of what kind of multilingual embeddings are the best for this task as this differs from one architecture to another, we notice that the more complex the model is the less accurate are embedding models trained from scratch $multi(skip\_gram)$ and $multi(sent\_ali)$ are and the more $multi(exp\_dict)$ reveals itself as performant. This is how we choose $multi(exp\_dict)$ to be used for a more complex model CNN-biGRU-Att. It is against our expectations to observe that $multi(skip\_gram)$ doesn't always outperform $multi(sent\_ali)$. This suggests that training the embeddings jointly over monolingual and multilingual constraints is not always beneficial as it was shown in the previous literature.

Results of churn detection using CNN-biGRU-Att model are shown in table 5.8. As we will explain next, this model architecture gives the best results for both monolingual and multilingual training and shows the true potential of multilingual training not only compared to state-of-the-art results. The first row shows the baseline results for training and testing on EN$_T$ data representing the state-of-the-art methodology (CNN) for churn detection defined in the work of Gridach et al. (2017). The results show that we outperform this baseline both monolingually and multilingually. If monolingually the difference of 0.38% is somehow marginal, the gain is more obvious using multilingual training with an F1-score of 85.88% which accounts for an increase of 2.03%. Our multilingual model is promising especially for German with an increase of 7.8% in F1-score. English also benefits with a slight increase of 1.65%.

To determine if the reported multilingual results are statistically significant with respect to monolingual results, we run the multilingual experiment training on $(EN + DE)_t$ and testing on EN_t and DE_t with 10 fold cross-validation 4 times. We use a One-Sample T-Test where the null hypothesis is that multilingual and monolingual results come from the same distribution. The tests of $(EN+DE)_t \rightarrow EN_t$ versus the mean of Churn Teacher and the mean of $EN_t \rightarrow EN_t$ show negligible p-values of $3.46e - 14$ and $3.42e - 14$ respectively with respect to $\alpha = 2.5\%$ which means we can reject the null hypothesis and say that the results are statistically significant. The same can be said for $(EN + DE)_t \rightarrow DE_t$ vs. $DE_t \rightarrow DE_t$ with a p-value of $3.01e - 12$.

Like CLDC, we try a multilingual setup training on EN_t only and test on DE_t. Unfortunately, this doesn't lead to significant improvement over training on German only or training on $(EN+DE)_t$ under the current architecture setup. So, we don't report those results.

To gain more insights into why the multilingual approach improves the test performance in German, let's take the following example: *"@MARKE das klingt gut zu den genannten Konditionen würde ich dann doch gern wechseln :)"*. This example which is not supposed to be churny is predicted as churny using German data only, while it is not detected churny according to the multilingual model. This is due to the fact that the German dataset lacks some important patterns that could be found in English and was naive to rely on the presence of *switch* keyword. On the other hand, multilingual approach is able to learn more complex patterns present only in resource-rich languages by taking advantage of the aggregation of both languages. As a matter of fact, there is a similar example in English: *"I want to switch to @BRAND already"* that portrays more or less the same pattern and which was learned successfully in English. The promise of multilingual embeddings is to enable transfer learning through transfer of annotation from high to low resource languages.

Model	Train	Test	Embeddings	F1-Score (%)	Precision (%)	Recall (%)
Fine Tuned MLP (FT-MLP)	EN_t	EN_t	*mono*	68.04	70.05	66.15
	$(EN + DE)_t$		*multi(pseudo_dict)*	**71.84**	70.68	**73.03**
			multi(exp_dict)	67.12	69.22	65.14
			multi(CCA)	70.89	70.52	71.28
			multi(sem)	68.12	70.80	65.64
			multi(sent_ali)	69.19	**72.55**	66.14
			multi(skip_gram)	70.91	72.35	69.52
	DE_t	DE_t	*mono*	55.46	59.93	51.61
	$(EN + DE)_t$		*multi(pseudo_dict)*	64.08	65.38	62.83
			multi(exp_dict)	62.13	66.53	58.27
			multi(CCA)	65.02	65.90	64.16
			multi(sem)	64.15	65.99	62.42
			multi(sent_ali)	**65.68**	66.01	**65.37**
			multi(skip_gram)	63.21	**66.6**	60.14

Table 5.5: Comparison of Detection in Social Media Results using FT-MLP

Model	Train	Test	Embeddings	F1-Score (%)	Precision (%)	Recall (%)
Multi-Filter CNN (MF-CNN)	EN_t	EN_t	*mono*	77.42	85.84	70.5
	$(EN + DE)_t$		*multi(pseudo_dict)*	79.31	83.51	**75.51**
			multi(exp_dict)	**79.43**	**85.56**	74.13
			multi(CCA)	76.76	82.52	71.76
			multi(sem)	73.37	80.0	67.76
			multi(sent_ali)	69.45	74.55	65.01
			multi(skip_gram)	78.22	86.33	71.5
	DE_t	DE_t	*mono*	58.58	68.22	51.33
	$(EN + DE)_t$		*multi(pseudo_dict)*	**69.03**	**77.32**	**62.34**
			multi(exp_dict)	66.81	74.40	60.62
			multi(CCA)	64.45	72.97	57.71
			multi(sem)	68.46	76.1	62.21
			multi(sent_ali)	65.57	70.29	61.45
			multi(skip_gram)	67.25	72.79	62.5

Table 5.6: Comparison of Detection in Social Media Results using Multi-Filter CNN (MF-CNN)

Model	Train	Test	Embeddings	F1-Score (%)	Precision (%)	Recall (%)
bi-GRU-Att	EN_t	EN_t	*mono*	**79.71**	81.48	**78.01**
	$(EN + DE)_t$		*multi(pseudo_ dict)*	76.86	77.73	76.02
			multi(exp_ dict)	78.28	82.15	74.76
			multi(CCA)	78.19	82.42	74.39
			multi(sem)	73.53	79.34	68.51
			multi(sent_ ali)	69.39	71.23	67.65
			multi(skip_ gram)	66.43	**90.44**	52.5
	DE_t	DE_t	*mono*	58.58	68.22	51.33
	$(EN + DE)_t$		*multi(pseudo_ dict)*	**69.87**	74.81	**65.54**
			multi(exp_ dict)	68.71	**75.74**	62.87
			multi(CCA)	65.39	74.12	58.50
			multi(sem)	66.26	73.24	60.5
			multi(sent_ ali)	67.17	71.89	63.03
			multi(skip_ gram)	60.63	67.24	55.2

Table 5.7: Comparison of Churn Detection in Social Media Results using Bidirectional GRU with Attention (bi-GRU-Att)

			Twitter Data		
Model	**Train**	**Test**	**F1-Score (%)**	**Precision (%)**	**Recall (%)**
Churn teacher	EN_T	EN_T	83.85	82.56	85.18
CNN-GRU-Att	EN_T	EN_T	84.23 ± 3.14	**87.70** ± 3.21	81.22 ± 4.08
CNN-GRU-Att	$(EN+DE)_T$	EN_T	**85.88** ± 2.36	85.85 ± 2.49	**85.94** ± 2.56
CNN-GRU-Att	DE_T	DE_T	66.69 ± 3.30	63.90 ± 5.80	70.44 ± 5.32
CNN-GRU-Att	$(EN+DE)_T$	DE_T	**78.09** ± 2.43	**78.62** ± 2.05	**77.72** ± 3.09

Table 5.8: Performance Comparison between multilingual and monolingual CGA Models and baseline (Gridach et al., 2017) of Churn Detection in Social Media

5.5.2 Chatbot Churn Results

The second part of the evaluation of multilingual embeddings for churn detection investigates their ability to transfer knowledge from one domain (social media) on which the model is trained to a different domain of application (chatbot conversations). The results for chatbot conversations are shown in table 5.9 which proves that they are comparable to previous social media results. This applies to both monolingual and multilingual modes of training. This proves that our model is able not only to capture classification patterns in social media tweets in both languages but also to generalize them to other domains of applications while keeping the same level of performance.

Moreover, we observe that multilingual training plays an important role in boosting the performance of monolingual models in this domain as well with an overall increase of 1.66% over both languages. Specifically, the model trained on $(EN+DE)_T$ and tested on EN_C is better than its monolingual counterpart trained on EN_T and tested on EN_C with an increase 2.33% in F1-score. However, the same cannot be said for the difference between multilingual and monolingual tested on DE_C as it exhibits a marginal drop 0.67%. One reason behind this could be the small number of conversation testing instances that we could collect and annotate given the limited time frame and budget. Their lack of variability could make them more similar in structure to the training tweets which makes them easy examples and not very representative of the population of conversation dialogues. Therefore, even a monolingual model would work well in this case.

5.6 Conclusion

In this chapter, we evaluate different multilingual embeddings on two variations of churn detection. We explore a plethora of different text classification architectures leveraging deep learning mainly Fine-Tuned MLP, Multi-Filter CNN, bi-directional GRU with attention and a novel combination of CNN and bi-GRU with attention. We investigate their performance using different multilingual embeddings and compare them to monolingual training mode. Our aim is to find for which text

Chatbot conversations					
Model	**Train**	**Test**	**F1-Score (%)**	**Precision (%)**	**Recall (%)**
CNN-GRU-Att	EN_T	EN_C	82.10	78.99	**85.45**
CNN-GRU-Att	$(\text{EN+DE})_T$	EN_C	**84.43**	**84.75**	84.18
CNN-GRU-Att	DE_T	DE_C	**74.25**	**74.14**	**74.32**
CNN-GRU-Att	$(\text{EN+DE})_T$	DE_C	73.58	73.45	73.72

Table 5.9: Performance Comparison between multilingual and monolingual CGA Models trained on Social Media and applied to Chatbot Conversations

classification models and languages the multilingual approach wins over the monolingual training. Experimental analysis reveals that there is always an average gain in performance in favor of the multilingual approach and this applies to all text classification architectures. The gain in performance is more pronounced the more under-resourced the language is, the less complex the architecture is and is mainly using fine-tuned embeddings models. This result highlights the universal facet of churn detection in social media and the usefulness of the multilingual approach to training high-performance models that can work consistently.

By applying the best model trained on churn detection in social media to chatbot conversations, we are able to show that our model can generalize and keep the same level of performance in case of multilingual training although the gap between monolingual and multilingual performance drops. Further investigation needs more data as this will more likely introduce cases where monolingual training alone is not enough and multilingual training reveals itself beneficent.

6

Cross-Lingual Event Detection

6.1 Overview

In this chapter, we design and implement an unsupervised event detection pipeline based on word embeddings which works in both monolingual and multilingual fashion. After pre-processing tweets collected using a streaming API, we divide the tweets into several small sub-windows. Then, we analyze tweets over sliding time windows. By focusing on one time window at a time, we reduce the complexity associated with trying to analyze all tweets together. After that, we extract candidate event triggers based on the analysis of their tweet and user frequency. Then, word embeddings (monolingual or multilingual) are incorporated to represent the triggers in a unified vector space and compute semantic similarities which are the input to a clustering algorithm. In the end, we apply further post-processing to filter out insignificant event clusters and to fine-tune the parameters. We define metrics for the qualitative and quantitative evaluation of this unsupervised methodology. The overall pipeline is depicted in figure 6.1.

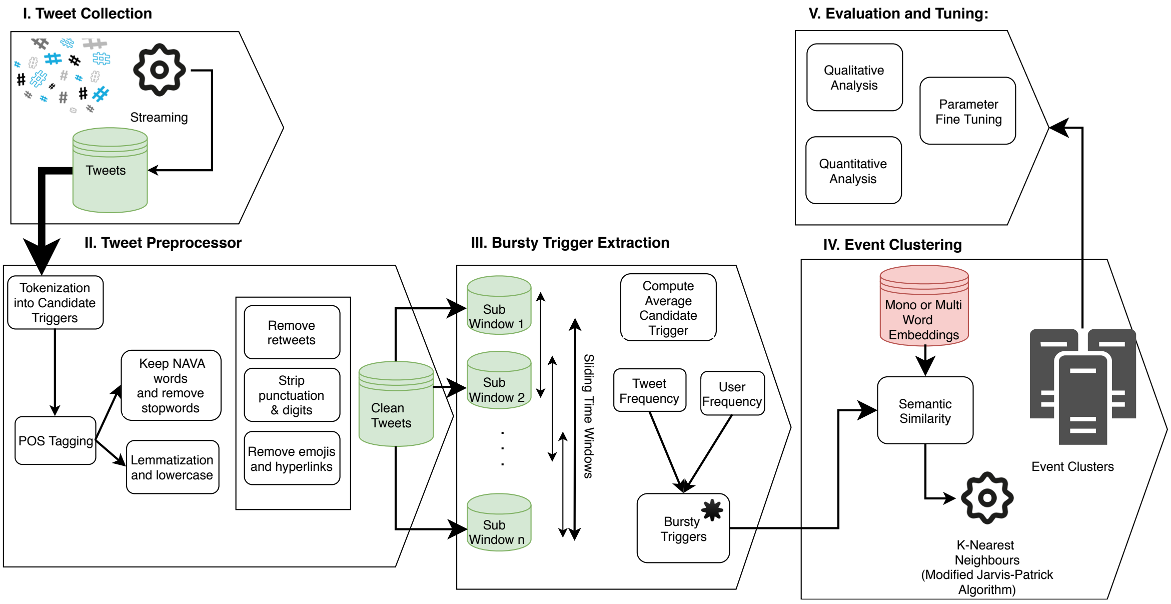

I. Tweet Collection
Streaming
Tweets
II. Tweet Preprocessor
Tokenization into Candidate Triggers
POS Tagging
Keep NAVA words and remove stopwords
Lemmatization and lowercase
Remove retweets
Strip punctuation & digits
Remove emojis and hyperlinks
Clean Tweets
III. Bursty Trigger Extraction
Sub Window 1
Sub Window 2
Sub Window n
Sliding Time Windows
Compute Average Candidate Trigger
Tweet Frequency
User Frequency
Bursty Triggers
IV. Event Clustering
Mono or Multi Word Embeddings
Semantic Similarity
K-Nearest Neighbours (Modified Jarvis-Patrick Algorithm)
Event Clusters
V. Evaluation and Tuning:
Qualitative Analysis
Quantitative Analysis
Parameter Fine Tuning

Figure 6.1: Event Detection Pipeline using Word Embeddings

6.2 Datasets

We evaluate our multilingual event detection pipeline on World Cup 2018. We choose this manifestation in particular because we expect that sporting events of this caliber to attract reactions of

	Hashtags /Followers
Official World Cup 2018	#WorldCup, #WorldCup2018, #WorldCup18, #FootballFever, #Russia2018, #Russia18, #FifaWorldCup18, #FIFAWorldCup, #Football, #soccergame, #världscupen, #CoppaDelMondo, #FIFA, #Cup, #Mundial, #WorldCupGame, #FIFAWorldCupGame, #coupedemonde, #coupedemonde18, #coupedemonde2018, #svjetskiKup, #VerdensMesterskab, #Weltmeisterschaft, #heimsmeistarakeppni, #MistrzostwaŚwiata, #CopaDoMundo, #CopaMundial
Team Code Tags	#bra, #sui, #crc, #srb, #ger, #mex, #swe, #kor, #bel, #pan, #tun, #eng, #pol, #sen, #col, #jpn, #rus, #ksa, #egy, #uru, #por, #esp, #mar, #irn, #fra, #aus, #per, #den, #arg, #isl, #cro, #nga
Official Accounts of FIFA and teams	@FIFAWorldCup, @RusFootballNews, @SaudiNT_EN, @Pharaohs, @UruFootballEN, @selecaoportugal, @SpainSoccerTeam@FRMFOFFICIEL, @TeamMelliIran, @equipedefrance, @Socceroos, @TuFPF, @DanishFooty, @Argentina, @footballiceland@HNS_CFF, @thenff, @CBF_Futebol, @SFV_ASF, @costaricasoccer, @FSSrbije, @DFB_Team, @miseleccionmx, @swefoot, @KORFootballNews, @BelRedDevils, @fepafut, @tunisiefootball, @pzpn_pl, @SenegalFootball, @FCFSeleccionCol
GameTags	#URUFRA, #BRABEL, #RUSCRO, #SWEENG, #URURUS, #KSAEGY, #IRNPOR, #ESPMAR, #DENFRA, #AUSPER, #NGAARG, #ISLCRO, #MEXSWE, #KORGER, #SRBBRA, #SUICRC, #JPNPOL, #SENCOL, #PANTUN, #ENGBEL

Table 6.1: Hashtags used to stream and search Twitter API

people of different nationalities and speaking different languages. Our goal is to test the ability of our automated approach to lifting the burden from journalists and entities which try to report on important game highlights covering multiple languages.

6.2.1 Tweets Streaming

We stream Twitter in real-time starting from the 13th of July 2018 (one day before the start of World Cup) till 19th of June (4 days after world cup is done) using Twitter Streaming API Tweepy. We filter by some of the official World Cup hashtags (for example "WorldCup18" and "Russia18", "FIFAWorldCup") and their translations to different languages like (CopaMundial, Weltmeistreschaft, coupedemonde2018", etc), in addition to team code hashtags (e.g. "ESP", "GER"), game tags (URUFRA, BRABEL, GERMEX, etc), Twitter usernames of FIFA and teams official accounts. Table 6.1 gives some more examples of the hashtags used. Due to the big number of hashtags, it was not possible to collect all tweets based on one streaming application. In order to reduce the streaming overload coming from the fact that the application processing time is not synchronized with tweets transmission rate, we create one application for each language independently. In the end, around 21,004,141 tweets were collected.

Figure 6.2 gives an idea about the distribution of the dataset per language. English and Spanish are the highest resource languages make up more than 70% out of the whole dataset. The reason why Spanish is a majority language besides English is due to a large number of teams from Latin America in addition to Spain making up 1/4 of qualified teams overall with largest fans around the world. On the other hand, other low-resource languages include Italian, German, Portuguese, French, Polish, Arabic, Russian, Persian and Korean. We are able to collect other languages like Danish, Icelandic, Japanese, Serbian and Swedish, but due to the very low frequency, we decide that they are not representative enough to be included. This data imbalance is kept deliberately in order to test our approach in the low-high resource languages scenario.

[0] We use python wrapper of Tweepy downloadable from https://github.com /tweepy/tweepy

Statistics	per Day	per Hour
Min	1,082	2,089
Mean	512,296	23,209
Max	1,166,619	209,290

Table 6.2: Distribution of Tweets per day and hour

Table 6.2 shows statistics of tweets per day and per hour. From figure **??**, we notice that there is a similar tendency in tweet distribution across languages: a sharp increase in the number of tweets in the mornings starting at 7 am reaching their apogee towards the middle of the day then either continuing till later in the day or next few hours in the next morning. To investigate the motives behind these peaks in the distribution, we plot the volume of English tweets for a particular day in 6.3(b). We can infer that before any game starts the number of tweets is relatively low. As a game begins, there is a surge of tweets posting both during the games and in the few hours following the games as marked by the peaks at hours 14, 17 and 20 which coincide with the ends of the games DEN-AUS, FRA-PER and ARG-CRO respectively (all times are GMT).

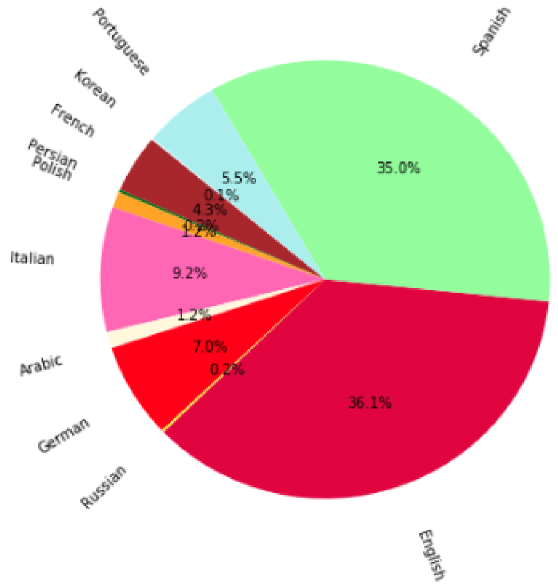


Figure 6.2: Tweet Distribution per language

6.2.2 Sub-Events Ground Truth

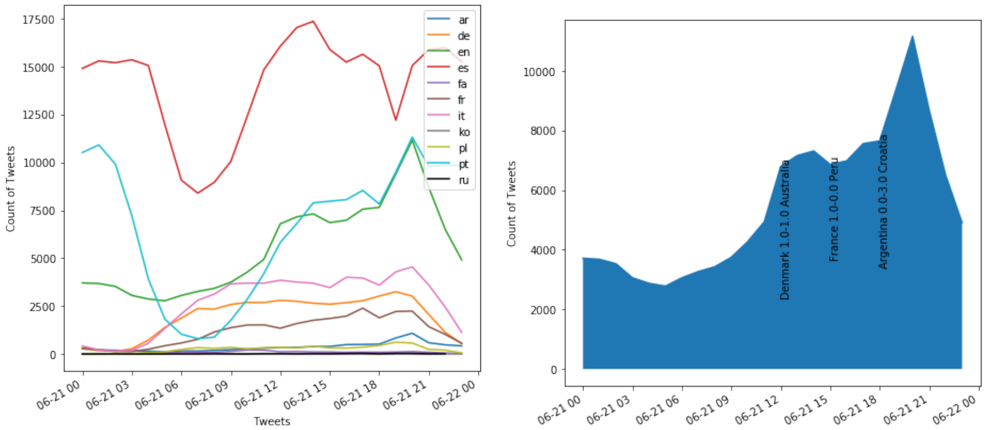
We use existing blog posts, news articles, and social media data to construct timelines for in-game events like scores, yellow/red cards, penalties, qualification/elimination. We consider this set of sub-events as our standard ground truth events and we match them to detected events to compare between multilingual and monolingual models. There is a total of 64 games with 32 teams, 169 goals scored, 219 yellow cards, 4 red card given which makes up an average of 2.6 goals, 3.5 yellow cards and 0.06 red cards per match.

6.3 Event Detection Methodology

6.3.1 Data Preprocessing

The Twitter dataset obtained consists of tweets each of which has the following attributes:

- **tweet_id**: unique identifier for each tweet
- **username**: the user that posted the tweet



(a) Per Language

(b) English Tweets, Game Correlation

Figure 6.3: Tweet Distribution for a particular day

- **retweets**: which is an integer counting indicating if the tweet is a reply to previous tweet
- **date**: which the timestamp the tweet was posted
- **text**: which the full text of tweet
- Other attributes like: favorites, geo, mentions, hashtags, permalink

Out of all these attributes, we are only interested in tweet_id, username, date, text, and retweets. We preprocess the tweets by breaking down them into tokens using SpaCy parser. This a scalable and extensible tool which performs tokenization, POS tagging and lemmatization at the same time with state-of-the-art accuracy (only tokenization is performed for languages non-covered by SpaCyThe covered languages by SpaCy are English, French, Italian, German and Spanish). Moreover, due to the noisy nature of Twitter data and particularity of event detection task, we do further preprocessing which includes:

- Stripping emojis, digits and URLs: otherwise, they will be detected as events by their own
- Stripping mentions of retweets RT and @username
- Removing punctuation, stopwords [1] and words whose length is less than 2 characters
- Dealing with Hashtags: removing Hash symbols but keeping hashtag content and splitting the compounds into words since our event detection pipeline only supports unigrams.
- Keeping only NAVA words: NAVA stands for Nouns, Adjectives, Verbs, and Adverbs. Those are more susceptible to be event triggers.
- Lemmatization and lower case conversion: to normalize terms

6.3.2 Bursty Trigger Extraction

In addition to the syntactic analysis performed by keeping only words with a relevant POS tag, we also perform filtering based on tweet and user frequency. Given the dynamic nature of tweets and their ever-changing focus and discussions that drift with time, we choose to focus on a specific time window at a time. The basic intuition behind combining both frequencies is that the more frequent a particular trigger is being mentioned and the more the users who post tweets about it, the more likely it refers to a hot popular event. "A high-quality feature for detecting bursts should yield higher weights for tokens that deviate significantly from normal behavior with respect to frequency or network density" (Buntain, 2014). These two measures make up the bursty weight of the trigger which is used to rank the triggers.

[1] The list of stopwords was obtained from NLTK and for rare languages they were extended from https://github.com/Xangis/extra-stopwords/blob/master/persian

The bursty weight of a trigger t within a time window tw is computed as follows:

$$\mathrm{w}_b(t, tw) = P_b(t, tw) * log_{10}(u_{t,tw}) \tag{6.1}$$

where $P_b(t, tw)$ is the bursty probability of a trigger t at tw and $u_{t,tw}$ is the number of users who post tweets containing t during tw. In other words, a trigger has higher bursty probability if its tweet frequency $f_{t,tw} \geq E[t|tw] + \sigma[t|tw]$ where $E[t|tw]$ and $\sigma[t|tw]$ are the expectation and standard deviation of frequency of tweets mentioning trigger t. This can be modeled as a binomial distribution and in the case of a large Twitter stream (N_{tw} is very large) approximated using Gaussian distribution:

$$\mathrm{P}_{f_{t,tw}} \sim \mathcal{N}(N_{tw}p_t, N_{tw}p_t(1 - p_t) \tag{6.2}$$

where N_{tw} is the number of tweets in the time window tw, p_t is the expected probability of tweets that contain trigger t in a random time window. Thus, the bursty probability $P_{t,tw}$ can be calculated as:

$$\mathrm{P}_{t,tw} = Sigmoid(10 \times \frac{f_{t,tw} - (E[t|tw] + \sigma[t|tw])}{\sigma[t|tw]}) \tag{6.3}$$

Based on that distribution, top K trigger are selected based on their bursty weights. The value of K is hard to define as a large K will add in more noise which will hurt precision and a small value would impact the recall. Based on a previous study (Li et al., 2012), a heuristic for the optimal value of $K = \sqrt{N_{tw}}$ which is used in this current work too.

6.3.3 Semantic Similarity and Event Clustering

After extracting bursty triggers, the next step is to cluster them in order to come up with event clusters. But before that, we need to first represent the trigger words as numerical features. For that purpose, we harness word embeddings W_e to project the triggers into a joint semantic vector space. In order to compute semantic similarity between every word pair, we use the cosine similarity between their vectors as in the equation:

$$sim(trig_a, trig_b) = \begin{cases} 1 - \dfrac{u_a . u_b}{\|u_a\|_2 \cdot \|u_b\|_2} & \text{if } trig_a, trig_b \in W_e \\ 0 & \text{otherwise} \end{cases} \tag{6.4}$$

Clustering based on semantic similarity is an additional filtering strategy as bursty triggers which cannot be clustered are considered as noisy and dropped from the list of triggers. Semantic similarity embedded on top of burstiness plays a crucial role to distinguish triggers exhibiting similar frequency patterns but belonging to different events, as we will show in the results section. For clustering, we use a non-hierarchical K-Nearest Neighbours clustering algorithm. It is a variation of the Jarvis-Patrick algorithm, which is depicted in algorithm 1. We choose this non-hierarchical clustering algorithm for its scalability, as it performs only one pass over the clusterable items. Basically, clustering consists of partitioning an undirected graph where vertices are triggers and edges between every two triggers a and b is $sim(trig_a, trig_b)$ as in equation 6.4. For each trigger, k-nearest neighbors are found. Then, for the two triggers to be in the same cluster, the following two conditions must apply:

- They are in each other k-nearest neighbors (i.e. trigger a is k-nearest neighbor of trigger b and vice versa).
- Among the k nearest neighbor, they share at least l common nearest neighbors

In addition to that, we add a parameter m which denotes the minimum number of triggers so that a collection of triggers can be considered as a relevant event.

6.3.4 Multilingual Extension

The multilingual approach consists of two possibilities as shown in 6.4. The idea is that a trigger may not be detected as bursty alone but is bursty when combined with other triggers from other languages. Thus, we investigate the following two approaches:

- **Multilingual Event Detection with Mono-lingual Triggers:** here we come up with bursty triggers for each language alone then we do event clustering the usual way explained throughout this chapter.

- **Multilingual Event Detection with Multilingual Triggers:** We perform only syntactic filtering and keep all candidate triggers after computing monolingual burstiness scores for later use. After clustering all candidate triggers, we keep only events that are bursty. The burstiness score for an event is simply the average of the burstiness scores for each of its triggers. This approach takes more time as there are more candidate triggers to be clustered which increases the time needed for clustering. For this reason, we mainly focus on the evaluation of the first approach.

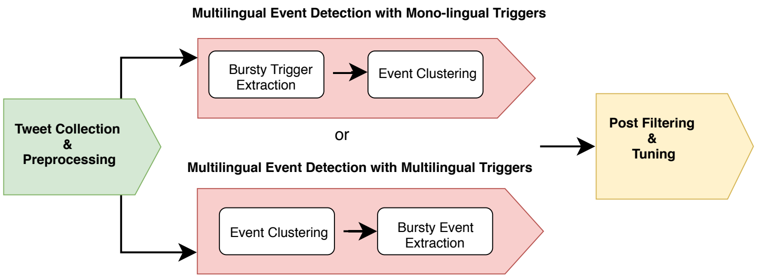

Multilingual Event Detection with Mono-lingual Triggers
Bursty Trigger Extraction
Event Clustering
Tweet Collection & Preprocessing
or
Multilingual Event Detection with Multilingual Triggers
Event Clustering
Bursty Event Extraction
Post Filtering & Tuning

Figure 6.4: Multilingual Event Detection Pipeline Approaches

6.3.5 Post Filtering and Tuning

Different studies have applied post filtering to remove noise inherent to Twitter data including measures like newsworthiness. However, we do not think it is relevant in our case since in our case, it might be possible to detect events that are not (or not yet) reported in news. Also, it is time-consuming to gather and select news that is relevant to the current context of the World Cup. The parameters that are fine-tune-able in this pipeline include clustering parameters k, l and m. To tune them, we rely on some evaluation metrics mainly the number of clusters detected and the number of triggers per clusters and only for some games. Due to lack of time, we could only try the following value ranges: $k \in [2, 3]$ and $l \in [1, 4]$. We notice that $k = 3$ gives clusters of better quality. On the other hand, we observe that as l increases, it becomes hard to find more clusters, so we relax the second condition in the clustering algorithm by choosing a low value $l = 1$.

6.4 Evaluation and Experiments

In order to evaluate the performance of the system and prove the gain that could be achieved with a multilingual approach, we design several monolingual and multilingual experiments as follows:

- $Ev_{Mono} - Tr_{Mono}$: Monolingual Event Detection with Monolingual Triggers
- $Ev_{Multi} - Tr_{Mono}$: Multilingual Event Detection with Monolingual Triggers

We run the above experiments on the time windows that contain the games of the world cup 2018 (i.e. meaning roughly 120 minutes). We use the same clustering parameters for comparing monolingual and multilingual models: $k = 3, l = 2$ and $m = 10$ Then, we follow two types of evaluation strategies to compare the different experiments:

- **Qualitative Evaluation:** we analyze the quality of the generated clusters by comparing the number of clusters, maximum, average and minimum size of clusters of monolingual and multilingual event detection.
- **Quantitative Evaluation:** we follow wavelet analysis to detect event peaks and their correlations with the occurrence of real-world events. The purpose of this evaluation is to show that we are not only able to detect numerous event clusters thanks to the multilingual approach but that those clusters correspond to significant signals. Our main hypothesis that we aim to investigate is if by combining using the multilingual approach many small peaks coming from monolingual tweets, we can get stronger peaks that are more likely to be attributed to a real-world event. For that purpose, we split our analysis into two stages:

Events	Keywords
Goals	"en_lose", "fr_joueur", "fr_score", "en_score", "es_golear", "fr_victoire", "fr_perdre", "en_win", "en_kick", "en_goal", "es_gol", "fr_but", "it_gol", "pt_golazo", "ar_هدف", "ko_목표", "de_tor", "pl_bramka", "fa_هدف","ru_Цель"
Yellow Cards	"en_card", "de_karte", "es_tarjeta", "fa_", "fr_carton", "it_carta", "ko_카드", "pl_karta", "pt_cartão", "ru_карта", "ar_بطاقة", "en_penalty", "de_strafe", "es_pena", "fa_مجازات", "fr_pénalité", "it_penalità", "ko_패널티", "pl_kara", "pt_pena", "ru_пенальти", "ar_جزاء", "ar_ضربة"
Red Cards	"en_red", "de_rot", "es_rojo", "fa_سرخ", "fr_rouge", "it_rosso", "ko_빨간", "pl_czerwony", "pt_vermelho", "ru_красный", "ar_أحمر", "de_beseitigung", "en_elimination", "es_eliminación", "fa_حذف", "fr_élimination", "it_eliminazione", "ko_제거", "pl_eliminacja", "pt_eliminação", "ru_устранение", "ar_إقصاء"

Table 6.3: Multilingual Keywords used for Fine Grained Event Detection

- *Without Ground Truth:* As a first step, we analyze for each event cluster its time-frequency distribution. In other words, we compute the total number of tweets which mention every trigger in the event cluster. Then, we perform wavelet analysis of those time series by computing for each event cluster the number of peaks above 1, 2 and 3 standard deviation(s) from the mean. It is important to note here that each event can have several peaks since it is not necessarily restricted to one timestamp. It is also common that users discuss the same event several times from different perspectives. Therefore, we do not report how many distinct events have peaks above 1, 2 and 3 standard deviation from the mean, but rather how many peaks exist per event cluster and then aggregate to have the average per game.

- *With Ground Truth:* We analyze the correlation of the detected peaks with real-time events to investigate if our approach performs better than random. For that, we define two types of correlations: time and keyword correlations. A detected peak is correlated in time with a true event if it occurs within a time window of duration t where the center is the timestamp of the occurrence of true event ($T_{true} - t/2 < T_{det} < T_{true} + t/2$). Our assumption is that if a peak is detected after a major event, it is more likely to be about that particular event. We restrict our analysis to the in-game time excluding half-time break as we observe a bigger tendency for peaks during half-time that could distort our analysis. The keyword correlation is proportional to the cosine similarity between the average of embedding representations of triggers of the detected and true event. We restrict our analysis to goals, yellow and red cards and define a representative set of keywords in multilingual languages as shown in table 6.3. We then compute recall, precision and f1-scores using two perspectives: goals vs. non-goals, events vs. non-events. We follow the equations explained in section 4.4.3 for computing precision, recall and f-score metrics. We define true positives, false negatives, and false positives as follows:

  * True Positive: a detected event that has both a time and keyword correlation with a true event (goal, yellow card or red card)
  * False Negative: a true event for which no detected event shares a time or keyword correlation
  * False Positive: a detected event that has a keyword correlation with a true event definition but doesn't have a time correlation.

For monolingual experiments, we use the same monolingual word embeddings described in section 3.2.1. For multilingual embeddings and since we are dealing with around 17 languages, we use already trained embeddings *multi_CCA* as described in section 3.2.1.2.

6.5 Results

In tables 6.4, 6.5, 6.7 and 6.8, we compare between the performance of event detection across different languages using qualitative and quantitative metrics without and with ground truth. For correlation with ground truth, we give a detailed analysis of two different views.

6.5.1 Qualitative Analysis

Table 6.4 reports on qualitative measures like the average number of clusters per game in addition to the minimum, average and maximum number of triggers per cluster. We notice that the number of clusters found using the multilingual approach is 5 times more and exceeds the average of the number of clusters per individual languages by 10.43 and the maximum number that can be reached using Spanish only by 6.54. We also notice that multilingual clusters are richer in content because they contain far more triggers as the average of the number of multilingual triggers accounts for 30% increase over the average over all monolingual triggers. There are also much more tweets that are related to the detected clusters in the multilingual case which is 6 times more than the average number of tweets for monolingual clusters. This is a positive indicator in the favor of the multilingual approach as by detecting a lot more clusters as a whole, chances that the detected clusters covering important events are higher. Other languages which are closer to multilingual performance include English and Spanish and this is normal since they are high-resourced.

Table 6.4: Qualitative Analysis Metrics of Event Detection across Different Languages

Languages	**#Clusters**	**Min Triggers**	**Avg Triggers**	**Max Triggers**	**# Tweets**
ar	2	10	11.58	21	805
de	2.25	11	14.58	18	1824
en	5.91	10	15.85	47	22176
es	6.26	10	**20.15**	79	17217
fa	0.33	7	7	21	134
fr	2.58	10	13.74	22	1330
it	2	11	12.66	15	1160
ko	0.5	12	6	12	20
pl	0.67	**13**	9.67	16	344
pt	3.14	10	14.15	26	6324
ru	0.5	12	7	16	72
multi	**12.8**	10	19.92	**208**	**32620**

6.5.2 Quantitative Analysis

6.5.2.1 Without Ground Truth

Like qualitative analysis, quantitative analysis without ground truth reveals gain of the multilingual approach. For 1, 2 and 3 standard deviations above the mean, multilingual clusters can be associated with way more peaks which are good candidates for events. For 1std, there are 41.18 more multilingual clusters than the average of monolingual clusters (roughly 3 times more). For 2std, there are 16.87 more multilingual clusters than the average of monolingual clusters (roughly 4 times more). While, for 3std, there are 3.81 more multilingual clusters than the average of monolingual clusters (roughly 3 times more). We notice that as we increase the number of standard deviations, the number of peaks decrease. This suggests that as we are looking for better quality peaks, we compromise their recall. This is normal and can be described as precision-recall trade-off inherent to event detection as we will show in the next sections.

Table 6.5: Quantitative Analysis Metrics of Event Detection without Ground Truth across Different Languages

Languages	**1std**	**2std**	**3std**
ar	15.45	7.27	2.82
de	13.75	5.75	2.5
en	35.09	11.07	3.61
es	45.74	13.83	3.17
fa	3.67	1.67	0.67
fr	25.0	8.0	2.58
it	8.0	1.5	0.0
ko	3.0	0.0	0.0
pl	6.67	1.0	0.67
pt	19.71	8.14	3.0
ru	3.25	2.0	0.5
multi	**57.48**	**22.34**	**5.86**

6.5.2.2 With Ground Truth

Fine-Grained Recall Table 6.6 shows recall scores for the different sub-event types: goals, red cards, yellow cards and their mean. Overall, we observe that multilingual approach wins on average over monolingual clusters by a margin of 24.33%, 25.12% and 11.47% for one, two and third standard deviations above the mean respectively. This fine-grained event detection analysis reveals that the multilingual combination of triggers is most effective in the case of goals with an increase of 36.64%, 39.01% and 16.43% over the average of monolingual goal recall and an increase of 7.44%, 5.36%, 2.04% over English for 1std, 2std and 3std respectively. Similarly, the multilingual approach performs better for yellow cards with an increase of 34.69%, 34.69% and 13.98% over average of monolingual goal recall and an increase of 4.47%, 4.47%, 0.39% over Spanish for 1std, 2std, and 3std respectively. The lower performance for Red Cards is explained by their low occurrence in the games and multilingual is no better than single language is due to the fact that correlation with red cards is found after goals and yellow cards.

	1std				**2std**				**3std**			
	G	**R**	**Y**	**M**	**G**	**R**	**Y**	**M**	**G**	**R**	**Y**	**M**
ar	66.49	0	30.3	32.26	58.05	0	30.3	29.45	26.45	0	7.58	11.34
de	25.0	0	12.5	12.5	16.67	0	12.5	9.72	0	0	6.25	2.08
en	80.88	2.27	51.29	44.81	54.92	2.27	51.29	36.16	27.39	0	10.13	12.51
es	82.33	**4.35**	53.3	46.66	68.67	**4.35**	53.3	42.11	34.44	0	17.21	17.22
fa	33.33	0	0	11.11	33.33	0	0	11.11	33.33	0	0	11.11
fr	78.39	0	34.51	37.63	53.29	0	34.51	29.27	33.41	0	3.75	12.39
it	33.34	0	12.5	15.28	16.66	0	12.5	9.72	0	0	0	0
ko	50.0	0	0	16.67	0	0	0	0	0	0	0	0
pl	22.22	0	16.67	12.96	22.22	0	16.67	12.96	22.22	0	0	7.41
pt	69.05	0	42.86	37.3	45.24	0	42.86	29.37	33.33	0	23.81	19.05
ru	27.5	0	0	9.17	16.25	0	0	5.42	10.0	0	0	3.33
multi	**88.32**	2.27	**57.77**	**49.45**	**74.03**	2.27	**57.77**	**44.69**	**36.48**	0	**24.2**	**20.23**

Table 6.6: [Quantitative Analysis with Ground Truth]: Recall Scores (%) for Fine Grained Event Detection including events types: G= Goals, R= Red Cards, Y= Yellow Cards and their mean

Goal Vs No-Goals View Table 6.7 shows results in terms of precision, recall, and f1-scores from the first perspective of Goal vs. No-Goals (true positive is when a detected event is correlated with Goal). We observe that the multilingual approach loses on precision while some languages mainly some low resource languages like Portuguese, Polish and Arabic have better scores in this regard. This is due to their low number of detected events which reduces the likelihood of having false positives. However, the higher recall for multilingual mode compensates for that loss and results in higher f1-scores. We observe that from 2 standard deviations and above, the gain in F1-score for multilingual vs. monolingual approach increases. Although for 1std, the multilingual mode is no better than the best monolingual model 66.14% vs. 81.69%, it is still better by 16% than the average over all languages. For 2std and 3std, the increase in F1-score gets bigger by 27.89% and 20.34% respectively.

	1std			**2std**			**3std**		
	F1	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**
ar	62.33	58.66	66.49	59.3	60.61	58.05	31.15	37.88	26.45
de	25.0	25.0	25.0	20.0	25.0	16.67	0.0	0.0	0.0
en	62.18	50.5	80.88	58.11	61.7	54.92	35.15	49.05	27.39
es	62.71	50.64	82.33	63.61	59.24	68.67	41.26	51.45	34.44
fa	33.33	33.33	33.33	33.33	33.33	33.33	33.33	33.33	33.33
fr	65.57	56.35	78.39	49.24	45.76	53.29	34.19	35.0	33.41
it	40.0	50.0	33.34	24.99	50.0	16.66	0.0	0.0	0.0
ko	50.0	50.0	50.0	0.0	0.0	0.0	0.0	0.0	0.0
pl	33.33	66.67	22.22	33.33	66.67	22.22	33.33	**66.67**	22.22
pt	**81.69**	**100**	69.05	55.4	**71.43**	45.24	42.1	57.14	33.33
ru	35.48	50.0	27.5	24.53	50.0	16.25	14.29	25.0	10.0
multi	66.14	52.87	**88.32**	**66.24**	59.94	**74.03**	**44.41**	56.74	**36.48**

Table 6.7: [Quantitative Analysis with Ground Truth]: F1-score, Precision and Recall Scores (%) for for Goal Vs No Goal View: F1= F1 score, P= Precision, R= Recall

Events Vs No-Events View Table 6.8 reveals that the same pattern observed in Goal Vs No-Goal view applies for the second perspective: events vs. no-events view (true positive is detected event is correlated with any sub-event type: goal, yellow or red card). In terms of precision, the multilingual approach has lower score when compared to some low-resource languages like Portuguese, Polish and Arabic. But, as we take more standard deviations above the mean the difference between multilingual and best monolingual model becomes smaller. However, multilingual does better than monolingual average by 21.3% and 39.9% for 2std and 3std respectively. Moreover, the higher recall for multilingual mode compensates for that resulting in higher f1-scores. We observe that from 2 standard deviation and above, the gain in F1-score for multilingual vs. monolingual approach increases. Although for 1std, the multilingual mode is no better than the best monolingual model 55.44% versus 56.68%, it is still better by 16% than the average over all languages. For 2std and 3std, the increase in F1-score gets bigger with 28.73% and 18.16% respectively.

As expected, event vs. non-event view carries lower results overall compared to goal vs. no-goal since the former includes red cards which hurt the recall scores. We also notice that precision suffers for the multilingual approach in this perspective since this is a more general perspective with a restricted definition of what is an event (either a goal, yellow or red card). It is possible that multilingual was able to detect other types of events but were not recognized as events which results in higher false positives rate.

6.5.3 Game Examples

Since it is impossible to check all clusters generated for all games, we choose some games, analyze them individually and look at their generated clusters. We provide some detected event clusters in appendix B.

6.5.3.1 Germany-Mexico Game

In tables 6.9, 6.10 and 6.11, we report on qualitative and quantitative performance of GER-MEX game. We not only notice that there are more multilingual clusters but better quality clusters as indicated by their average number of triggers bigger than the average for monolingual clusters 20.95 vs. 17.49. We also observe that the number of tweets in which multilingual triggers are mentioned is 3.5 times bigger. This suggests that the multilingual approach is better than the aggregation of all languages as the triggers found by multilingual approach span over more multilingual tweets than

	1std			**2std**			**3std**		
	F1	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**
ar	46.32	82.08	32.26	43.31	81.82	29.45	18.86	56.06	11.34
de	20.0	50.0	12.5	16.28	50.0	9.72	3.84	25.0	2.08
en	51.52	60.6	44.81	47.7	70.07	36.16	20.94	64.28	12.51
es	**56.4**	71.29	46.66	54.03	75.36	42.11	27.72	71.01	17.22
fa	16.66	33.33	11.11	16.66	33.33	11.11	16.66	33.33	11.11
fr	45.94	58.95	37.63	37.99	54.1	29.27	19.99	51.67	12.39
it	26.51	**100**	15.28	16.28	50.0	9.72	0	0	0
ko	25	50	16.67	0	0	0	0	0	0
pl	21.7	66.67	12.96	21.7	66.67	12.96	13.34	66.67	7.41
pt	54.33	**100**	37.3	43.75	**85.71**	29.37	30.08	71.43	19.05
ru	15.5	50.0	9.17	9.78	50.0	5.42	5.88	25.0	3.33
multi	55.44	63.08	**49.45**	**56.68**	77.45	**44.69**	**32.46**	**82.12**	**20.23**

Table 6.8: [Quantitative Analysis with Ground Truth]: F1-score, Precision and Recall Scores (%) for for Event Vs No Event View: F1= F1 score, P= Precision, R= Recall

the addition of monolingual tweets. Moreover, there are more chances of finding fine-grained events as revealed by quantitative analysis with Ground Truth.

In order to visualize the temporal information of the tweets related to the detected clusters, we plot in figure 6.5 the total count of tweets that mention each of the triggers in one detected cluster. We notice that there is not only a keyword correlation (with triggers like en_win, en_victory, es_ganar, es_eliminar and so on) but also a peak towards the end of the match announcing the victory of Mexico and the defeat of Germany and investigating the possibility of elimination of Germany.

	# Clusters	**Avg Triggers**	**Max Triggers**	**Min Triggers**	**# Tweets**
de	2	16.0	17	**15**	1423
en	6	**21.83**	35	10	38628
es	11	14.64	26	10	22154
multi	**19**	20.95	**68**	10	**69835**

Table 6.9: Qualitative Analysis of GER-MEX Game

	1std	**2std**	**3std**
de	12	7	4
en	25	11	7
es	41	11	9
multi	**44**	**12**	**8**

Table 6.10: Quantitative Analysis without Ground Truth for GER-MEX Game

	1std			**2std**			**3std**		
	G	**R**	**Y**	**G**	**R**	**Y**	**G**	**R**	**Y**
de	0	0	3	0	0	0	0	0	0
en	0	0	3	0	0	0	0	0	0
es	1	0	3	0	0	0	0	0	0
multi	**1**	**0**	**3**	**0**	**0**	**2**	0	0	0

Table 6.11: Quantitative Analysis with Ground Truth for GER-MEX Game

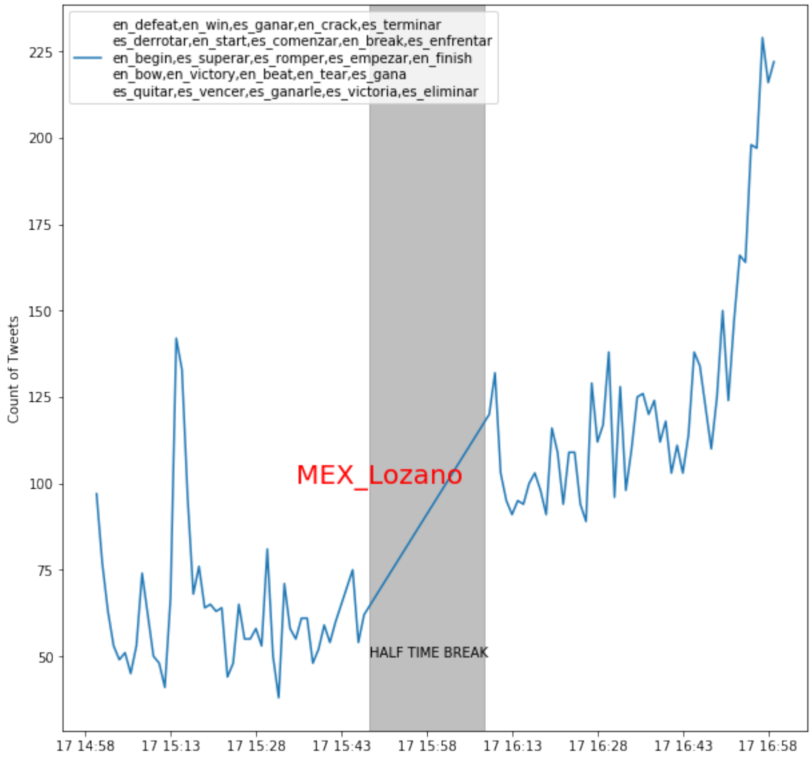


Figure 6.5: Time Series of Goal Event in GER-MEX game and its correlation with ground Truth

6.5.3.2 Portugal-Spain Game

In tables 6.12, 6.13 and 6.14, we report on qualitative and quantitative performance of POR-ESP game. Like GER-MEX, we notice a large gain in favor of the multilingual approach both qualitatively and quantitatively. There are more event clusters of bigger size and a higher number of correlations of goals detected at different standard deviation levels.

In figure 6.6, we show temporal peaks of tweets related to an event cluster that has a higher keyword correlation with goals of Ronaldo such as es_ronaldo, es_delantero, es_cristiano, es_mascherano, es_entrenador, es_tecnico, es_futbolista, es_jugador. We notice that there are more peaks well correlated with the times of occurrence of the three goals of Ronaldo detected by one event cluster.

	# Clusters	Avg Triggers	Max Triggers	Min Triggers	# Tweets
en	12	15.25	31	10	54759
es	6	**24.5**	**53**	13	22146
pt	3	18.0	22	**14**	7034
multi	**26**	18.88	51	10	**89778**

Table 6.12: Qualitative Analysis of POR-ESP Game

	1std	2std	3std
es	29	11	1
pt	9	6	3
en	27	9	4
multi	**53**	**21**	**9**

Table 6.13: Quantitative Analysis without Ground Truth for POR-ESP Game

	1std			2std			3std		
	G	**R**	**Y**	**G**	**R**	**Y**	**G**	**R**	**Y**
es	6	0	2	2	0	**1**	1	0	0
pt	0	0	0	0	0	0	0	0	**1**
en	4	0	1	1	0	0	1	0	0
multi	**6**	**0**	**2**	**4**	**0**	0	**2**	0	0

Table 6.14: Quantitative Analysis with Ground Truth for POR-ESP Game

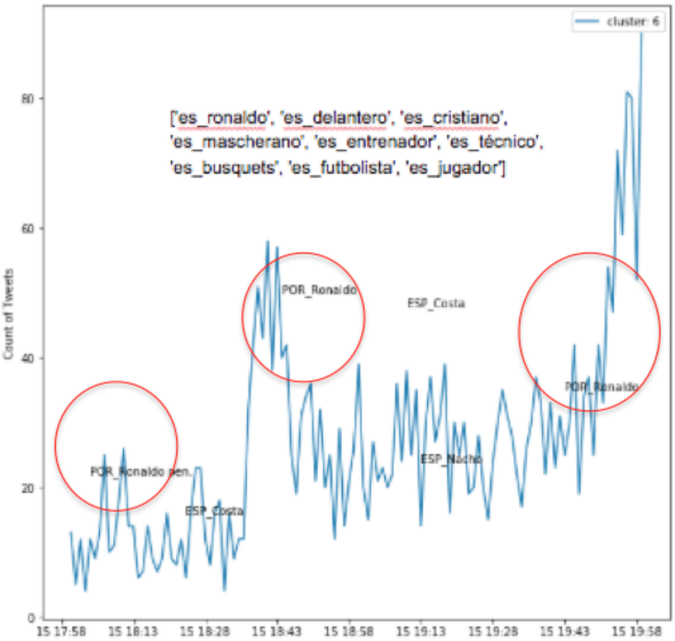


Figure 6.6: Time Series of Goal Event in ESP-POR game and its correlation with ground Truth

6.6 Conclusion

All in all, we elaborate a framework for cross-lingual event detection done in an unsupervised fashion. Our approach relies on the aggregation of bursty triggers from different languages and on clustering them using word embeddings. Results show that there is a gain in favor of the multilingual approach in both qualitative and quantitative analysis and from two different perspectives. Although precision suffers sometimes, recall is always in favor of the multilingual approach which is more important in the type of applications our system will serve. This recall/precision tradeoff is even more pronounced in event vs. no-events perspective. This system can serve as an alarm system to detect anomalies assuming they will be human analysts to decide on which sub-events are more relevant in a real-time scenario. In this chapter, we apply this framework to retrospective event detection, but the same can be extended to real-time detection of any event type.

Algorithm 1 Modified Jarvis-Patrick Clustering Algorithm

```
Require: $T$ is the set of triggers and $sem\_sim$ is the semantic similarity between triggers, $k$ is the
    number of neighbor and $min\_cluster\_triggers$ is the minimum number of trigger elements in
    the cluster

function FindKnnClusters($T$, $sem\_sim$, $n$, $min\_cluster\_triggers$)
     Let $clusters$ the set of output event cluster
     Build undirected graph G with the set of vertices $V = T$
     Let $k\_neighbor$ be the dictionary of neighbor for each vertex in $V$
    for each $v_i$ in $V$ do
         Let $neighbor$ be the set of neighbor for $v$
        for $v_j$ in $V$ where $ij$ do
             Add $sem\_sim(v_1, v_2)$ to $neighbor$
        end for
         Sort neighbor and select top K neighbor
         Add neighbor to $k\_neighbor[v_1]$
    end for
    for each $v_i$ in $V$ do
        for each $v_j$ in $V$ where $ij$ do
            if $k\_neighbor[v_1] \cap k\_neighbor[v_2] \geqslant k$ then Add edge $E(v_i, v_j)$
            end if
        end for
    end for
    for each $comp$ in connected components in $G$ do
        if number of vertices in $comp \geqslant min\_cluster\_triggers$ then Add to $clusters$
        end if
    end for
```

7 Conclusion and Future Work

In this thesis, we put in place a systemic multi-dimensional comparative analysis of multilingual embeddings on several supervised and unsupervised intrinsic and downstream tasks. We provide a unified approach for training across languages leveraging different multilingual embeddings methods and an end-to-end benchmark for their evaluation against their monolingual counterparts. The embeddings covered in our analysis span a diverse spectrum of methodologies covering those trained from scratch, those fine-tuned on top of monolingual embeddings and those learned jointly with the task. We revisit an existing benchmark for Cross-lingual document classification. We adapt several neural networks to this hierarchical nature of documents. We test both in an imbalanced data scenario with English being the most dominant language and in a low data regime and witnessed the consistent gain of multilingual approach especially for low-resource languages. Relying on the same methodology, we define and test a novel framework for cross-lingual churn detection and achieve state-of-the-art results for both languages. In the end, we define a new framework for unsupervised cross-lingual event detection based on multilingual clustering of bursty triggers and tested our approach on a freshly collected worldwide manifestation. In what follows, we conclude on the main contributions, findings, encountered challenges and venues to explore in future work for each component of this thesis.

7.0.1 Creation of Multilingual Embeddings

Existing work evaluates against different tasks using several bilingual embeddings. In this thesis, we are more interested in multilingual embeddings where more than two languages at a time are aligned to the same space. Following our related work investigation, we base our analysis on only supervised methodologies as the performance of unsupervised methodologies did not prove promising. We pick a set of representative embeddings from the two different families: fine-tuned and trained from scratch. For some methodologies such as $multi(CCA)$ and $multi(skip\_gram)$, embeddings for the languages of our interest: English, German, French and Italian are obtained directly. On the other hand, the multilingual extension for other embeddings types is generated from scratch. For fine-tuned embeddings with SVD, we generate two versions paying attention to the role of dimensionality reduction and the translation pairs used.

We close up this chapter with a qualitative analysis to compare the performance of different embeddings in a translation task, cross-lingual nearest neighbours and check their vocabulary coverage. Although fine-tuned embeddings using SVD with $expert\_dict$ outperformed all other embeddings in terms of quality of translation and coverage, the other models which lagged behind in this intrinsic evaluation are not necessarily the ones that performed the worst in all downstream tasks later on. In other words, our investigation brings us to think that there is no direct correlation between intrinsic and extrinsic evaluation. Our results confirm the previous literature finding that this is one of the main limitations of working with multilingual word embeddings. While they can be good at modeling the cross-lingual conceptual meaning of words as in tasks like word translation, word similarity, and nearest neighbor, they do not necessarily have the same ability of capturing the relationships between the words, word order and other textual characteristics crucial for good results on intrinsic evaluation to reflect directly on extrinsic evaluation. For those reasons, we not only explore the performance of those embeddings as they are but also their performance as they trained to learn the compositional representation of documents and sentences in later chapters.

While this evaluation tries to cover a representative set of methodologies, investigating other sub-approaches could reveal new results. In generating embeddings in this thesis, all languages are treated equally except English which is the target language. This is a naive assumption that doesn't take into consideration the syntactic similarities and differences between languages which could impact the quality of alignment. For example, aligning Chinese or Arabic to English is not guaranteed to be of the same quality of aligning German to English. To investigate this, we hope to embed syntactic dependencies into the framework of the generation of multilingual embeddings in future work. Another limitation of multilingual embeddings is their inability to properly deal with polysemy. This issue is more serious in the case of multilingual embeddings as translation pairs used to learn the alignment assume a one to one mapping which is not always true. Investigating strategies to circumvent this issue is another potential extension in future work.

7.0.2 Cross-Lingual Document Classification

Existing benchmark for Cross-Lingual Document Classification evaluates the performance of bilingual embeddings after feeding them to an averaged perceptron algorithm. The main motivation is to show their role in transfer learning from one language to another. We follow the same coarse-grained dataset definition and extend the approach to training over many languages at the same time and evaluate its performance against monolingual training. We investigate both embeddings obtained independently and those trained jointly and experiment with text classification architectures at different levels of deepness: Fine-Tuned MLP, Multi-Filter CNN, and Hierarchical Neural Networks.

We experiment with two data regimes scenarios: imbalanced multilingual dataset with one dominant language and low data regime balanced over languages. In the case of an imbalanced multilingual dataset, our results show that there is a gain in favor of multilingual training especially for low resource languages mainly Italian, French, and German. Although those results are consistent across architectures, they are more pronounced the less deep an architecture is. Results of Multi-tasking training embeddings using sentence alignment and document classification tried in low data regime show a bigger gap between multilingual and monolingual performance for all languages. Therefore, we can only recommend training using multilingual embeddings in the presence of valid use cases. The promise of multilingual embeddings is their ability to compensate for the weaknesses of single language specific models in the absence of enough training instances (e.g. missing instances for certain classes) to reach maximum performance in one language by its own.

While training over the aggregation of all languages, we came across issues like false cognates (words that are spelled the same but belong to different languages). Our attempt to address it is by introducing a language prefix to each word. This is simplistic as it assumes that we know the language to which each word belongs to. In future work, it could be worth investigating a more principled solution relying for example on the multi-lingual context to distinguish the cognates. Other possibilities for future work include exploring other strategies for multi-tasking.

7.0.3 Cross-Lingual Churn Detection

Using the same benchmark developed for CLDC, we investigate the performance of two variations of churn detection in social media and chatbot conversations. In addition to Fine-Tuned MLP and Multi-Filter CNN, we explore bi-GRU-Att and an exclusive combination of CNN and biGRU-Att. Our multi-dimensional analysis reveals interesting trends across languages, embeddinsg families and text classification architectures that are consistent with the findings CLDC. We are able to prove that churn detection is a universal task that can be trained efficiently multilingually on all languages irrespective of the architecture used. The gain is more pronounced in the case of low-resourced languages and the less complex the model is. This has huge implications as it simplifies training a complementary cross-language model without the need for a complex model, leveraging the simplest architecture and the only available data instances. Our main contribution is a state-of-art architecture for cross-lingual churn detection using CGA architecture which is a novel task. In future work, we hope to investigate the performance on under-resourced languages as the company is more interested in covering dialects of its Swiss customers. It would also be interesting to validate our conclusions on bigger datasets especially for churn detection in chat-bot conversations.

The radar chart in figure 7.1 shows the impact of different variables on the performance of multilingual embeddings. There are four variables two of which are related to the way the embeddings are generated (degree of supervised of embeddings and linearity of embeddings methodology) and the nature of the dataset and classification model (language resourcefulness and classification model complexity). In sum, we can conclude multilingual embeddings bring greater value the more supervised the methodology of their generation is, the less resourceful the languages it is applied on are and the less complex the classification model.

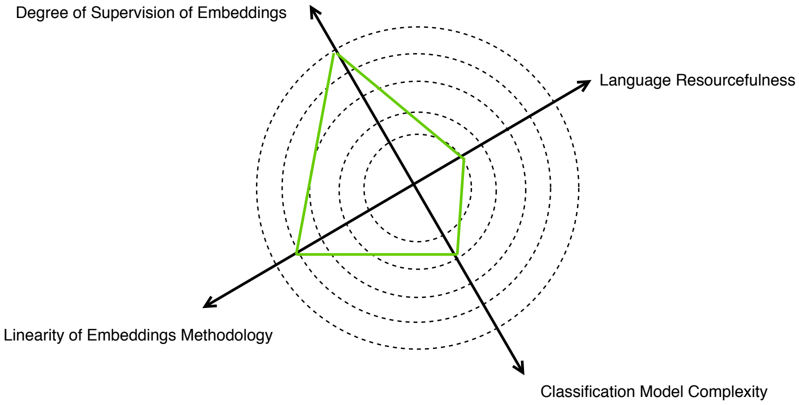

Degree of Supervision of Embeddings
Language Resourcefulness
Linearity of Embeddings Methodology
Classification Model Complexity

Figure 7.1: Multi-Variate Analysis of Performance of Multilingual Embeddings on Text Classification

7.0.4 Cross-Lingual Event Detection

Most existing works on multilingual event detection rely on language agnostic models featuring only burstiness measures. The contribution of this thesis is the design of a framework where multilingual embeddings are employed on top of burstiness as an additional filter that bridges the gap between languages and enables clustering of multilingual triggers. By validating this methodology on a multinational manifestation, our results show a gain in favor of the multilingual approach both qualitatively and quantitatively. Using correlation with ground truth consisting of three fine-grained events, we are able to show that the gain is not random as we observe that multilingual clusters are better related to real-world events. The main contribution of this work is its ability to show how the aggregation of monolingual signals results not only in more but also stronger multilingual signals.

The first venue to explore in future work is to work on crowdsourcing sub-event annotation of the collected tweets for a more accurate evaluation. One of the limitations of this work is its use of unigrams only. Segmentation to n-grams can be very useful for event detection, as it can convey more concrete information leading to clearer interpretability and a better precision. Given the nature of event detection task, many triggers are in the form of named entity and proper nouns which are not always covered by the multilingual embeddings of unigrams. Our focus in this thesis is to evaluate the performance of a multilingual approach leveraging multilingual word embeddings. Obtaining multilingual embeddings for n-grams is expensive given the large number of languages targeted. In future work, a potential extension is to investigate efficient solutions for coming up with the multilingual representation of bigrams and trigrams and to adapt them to the current domain of application (sport events in this case).

It could be also worth investigating with other variations for generating word embeddings. However, we expect that the impact of multilingual phrase embeddings on improving the performance will be more important than trying with better methodologies for multilingual word embeddings. Another venue to be explored in future work is the performance of this framework when extended to other domains of applications. These include industrial use cases to detect anomalies in the behavior of multinational customers. The usefulness of a multilingual event detection framework, in this case, is to investigate in an unsupervised way based on the frequency of semantically correlated triggers for example customer reactions to the release of a new product or service.

A Multilingual Documents Nearest Neighbours

Document	Class	Nearest Neighbours	
"New york 3 décembre reuter wall street a clôturé en baisse de 1,21 mardi l'indice dow jones des 30 industrielles abandonnant 79,01 points à 6.442,69 des ventes programmées ont lourdement pesé sur la cote en fin de journée ce qui a eu pour effet de déclencher à 15h50 locales 20h50 gmt la procédure du coupe-circuit qui limite les ventes liées à l'arbitrage ce mouvement n'a semble-t-il touché que les valeurs vedettes et sur toute l'ensemble de la cote on comptait plus de hausses 1.294 que de baisses 1.194 à la clôture."	Markets	English	"Argentine bonds put in a strong performance monday helped by general flows into the latin american region although activity was still fairly slim the market rose with the whole region " said a trader at a large foreign bank the market began to rise last week after several u.s. indicators showed moderating u.s. growth with market friendly argentine indicators released late friday and this session bonds were given an additional boost argentina 's frb due 2005 rose 3/4 to 83-6/8 while the local dollar-denominated bocon previsional 2 due 2001 rose 0.55 to 119.90"
		German	"Zürich reuter die conf-futures und der smi index haben am mittwochnachmittag mit leichten kursverlusten auf die veröffentlichung der us-konsu -menten-preise reagiert der juni-conf stand um 14.40 uhr bei 118,42 prozent gegenüber 118,49 prozent unmittelbar vor der veröffentlichung der smi index sank auf 4535,3 punkte von 4543,7 ... "
		Italian	New "York 7 feb reuter wall street e prevista al rialzo in apertura dopo che i dati usa sul lavoro hanno fatto schizzare i t-bond gli analisti notano che il numero degli occupati non agricoli e risultata superiore al previsto mentre la componente salari orari e inferiore alle attese e lascia preve -dere il proseguimento di un periodo di bassa inflazione alle 14,50 il futures sull'indice s p ..."

Table A.1: Cross-Lingual Documents Nearest Neighbours

Document	Class	Nearest Neighbours	
"Roma 8 luglio reuter il ministro degli esteri lamberto dini ha incontrato al quirinale il presidente della repubblica oscar luigi scalfaro lo rende noto un comunicato del quirinale che come di consueto non da dettagli sul contenuto dei colloqui"	Government	English	"These are the leading stories in the egyptian press on thursday reuters has not verified these stories and does not vouch for their accuracy al-ahram foreign ministers of the damascus declaration states will discuss egypt 's proposal on setting up arab free zones or an arab common market the group includes syria egypt and the six gulf arab countries egypt to propose at the u.n. earth summit imposing a tax to finance a world fund for permanent development the international monetary fund expects growth rate in egypt to rise to 5.5 percent al-akhbar the governors council chaired by prime minister kamal ganzouri decides to keep the value of state farm land leases unchanged information minister safwat el-sherif says the new law which ends the perpetuated leases will be enforced in october the new bill has stired the anger of landtenants trade was worth 454 million pounds at the stock market on wednesday al-gomhuria public minister atef obeid says in new york that 30 state firms will be offered for sale by the end of 1997 six people charged with selling exams of the general secondary school certificate will be tried before the state security court 1 3.395 pounds – cairo newsroom"
		French	"Paris 2 juin reuter la composition du gouvernement sera annoncée mercredi ou jeudi a déclaré lundi sur lci le porte-parole du ps françois hollande interrogé sur le fait de savoir si la composition du gouvernement serait annoncée mercredi il a répondu “ mercredi ou jeudi ” /jmb"
		German	"Ramallah reuter aus protest gegen die behinderung seines treffens mit dem früheren israelischen ministerpräsidenten schimon peres hat palästinenser-präsident eine berater-konferenz mit israel abgesagt der palästinensische abgeordnete hassan asfur sagte der nachrichtenagentur reuter am donnerstag betroffen sei eine unterredung der verhandlungsbeauftragten des präsidenten dschamil el tarifi und oren schahor von der israelischen regierung israel hatte arafat nach palästinensischen angaben am donnerstag einen hubschrauber-flug zu einem treffen mit peres im westjordanland verweigert die begegnung wurde deswegen in den gaza-streifen verlegt"

Table A.2: Cross-Lingual Documents Nearest Neighbours (Cont.)

B

Event Clusters Examples

Mode	Event Triggers	Event Description
mono_de	<de_worldcup, de_startelf, de_dfb, de_mundial, de_nationalmannschaft, de_mannschaft, de_fußball, de_fussball, de_copa, de_fifa, de_spieler,de_cup, de_alemania, de_fifaworldcup, de_minute>	Germany World Cup Game
	<de_stehen, de_bereiten, de_live, de_sehen, de_wünschen, de_finden, de_beginnen, de_geben, de_schauen, de_zdf, de_zeigen, de_erfolgen>	The first match for Germany
mono_en	<en_cup, en_win, en_attack, en_competition, en_match, en_draw, en_championship, en_loss, en_winner, en_upset, en_challenge, en_tie, en_defend, en_tournament, en_injury, en_surprise, en_clash, en_lose, en_victory, en_champions, en_defense, en_champion, en_fight, en_defeat>	Winning of Mexico & Defeat of Germany
	<en_football, en_footballer, en_play, en_hear, en_kick, en_loud, en_goal, en_run, en_player, en_game, en_penalty, en_soccer, en_freekick, en_listen, en_goalkeeper>	Goal of Mexico
mono_es	<es_golear, es_minuto, es_ganar, es_seleccion, es_derrotar, es_gol, es_selección, es_golazo, es_fútbol, es_ganador, es_fifa, es_campeón, es_copa, es_enfrentar, es_marcar, es_anotar, es_gana, es_fifaworldcup, es_vencer, es_cup, es_ganarle, es_football, es_futbol, es_concacaf, es_deportivo, es_empatar>	Goal of Mexico
	<es_jugador, es_jugar, es_éxito, es_delantero, es_futbolista, es_debut, es_zague, es_portero, es_arquero, es_participación, es_participar, es_debutar, es_entrenador, es_actuación>	Game at the first days of World Cup
	<es_herrera, es_moreno, es_salcedo, es_ayala, es_hernández, es_lozano, es_osorio, es_guzmán, es_bolaños>:	Main players in Mexico Team: Lozano: goaler and Herrera, Moreno: yellow cards holders
	<es_oportunidad, es_sorprender, es_tremendo, es_espectacular, es_increíble, es_tecnología, es_impresionante, es_calidad, es_excelente, es_sorpresa>	Mexican Joy after winning game

Table B.1: Sub-Event Related Clusters Examples for GER-MEX Game

Mode	Event Triggers	Event Description
multi	<es_carlos, de_mex, de_mundial, de_vela, es_javier, de_herrera, de_moreno, en_ochoa, es_mex, de_hernandez, es_juan, en_carlos, es_hernández, en_lozano>	Goal of Mexico
	<en_champions, en_championship, es_cup, en_cup, en_tournament, es_ganador, en_title, es_título, en_champion, en_winner, es_football, es_campeón, de_football, es_copa>	Mexico winning
	<en_germans, es_mexicanos, es_alemán, en_americans, en_fuck, en_shit en_damn, es_méxico, en_german, en_mexicans, en_fucking, en_people>	Angry Germany loosing
	<de_nationalhymne, de_singen, es_camiseta, en_song, de_hymne, en_music, en_flag, en_sing, es_bandera, es_tocar, es_cantar, en_anthem, es_vestir, es_coser, es_himno es_canción>	German national hymn
	<en_defeat, en_win, es_ganar, en_crack, es_terminar, es_derrotar, en_start, es_comenzar, en_break es_enfrentar, en_begin, es_superar, es_romper, es_empezar, en_finish, en_bow, en_victory, en_beat, en_tear, es_gana, es_quitar, es_vencer, es_ganarle, es_victoria, es_eliminar, es_acabar>	Mexico Winning Vs. Germany Loosing
	<es_ojalá, en_awesome, en_funny, en_wonderful, en_amazing, en_updates, en_prediction, en_experience, en_wow, en_like, en_passion, en_fantastic, en_review, es_emoción, en_deserve, en_talent, es_arquero, en_player, es_portero, en_footballer, es_jugador, es_técnico, es_delantero, es_futbolista, en_coach, es_entrenador>	Praise of talents of players

Table B.2: Sub-Event Related Clusters Examples for GER-MEX Game (Cont.)

C

Multilingual Library: Readme File

This software deliverable for this thesis consists of a python library for inducing and evaluating multilingual embeddings in different applications including their qualitative analysis. The objectives of this library are to provide an end to end benchmarking system where given cross-lingual datasets for document classification, churn detection or event detection and multilingual embeddings on any set of languages, it performs adapted preprocessing, trains/fine-tunes multilingual models and return evaluation results against monolingual experiments.

C.0.1 Installing Requirements:

We provide a Dockerfile to build and run an image that has all required packages for executing all programs in this library.

C.0.2 Code Walkthrough:

Code for reproducing the results is organized into the following sub-directories:

C.0.2.1 [MultiEmb]: Generation of Multilingual Embeddings

This sub-directory includes code for generating embeddings using offline methodologies: multi(pseudo_dict), multi(exp_dict) and multi(sem). It also contains code for training and evaluating Multi-Tasking methodology on CLDC.

Offline Methodologies: The main script to run main.py with option –dict-type= "expert" for generating embeddings using ground truth dictionaries and with option –dict-type="psdo" for generating embeddings with pseudo dictionary. Other options include what src languages –src-lang (separated by undescore) and which target language –trg-lang to use, whether to use dimensionality reduction or not in –dim-red and the path of monolingual embeddings to use in –mono-emb-path.

Run compute_alignments.py for applying already learned alignment directly on monolingual embeddings and for saving the whole in one file.

Translation Matrices and Visualization: Run compute_precision_translation.py to get the precision @1 and @5 of the translation matrices on any source and target languages against a test dictionary.

Multi-Tasking: Run main.py for multilingual experiments adapting –train-langs and –test-langs as needed. For monolingual experiments, the script main_mono.py is to be used.

C.0.2.2 [CLDCEval]: Cross-Lingual Document Classification using Multilingual Embeddings

The first goal consists of benchmarking different multilingual embedding models and their application to document classification across languages.

The main script to run is "main.py" which after performing data preprocessing, converts each word to each corresponding vector using either monolingual or multilingual embedding model depending on the mode of evaluation chosen, then runs classification of choice including simple MLP, linear SVM and multi filter CNN. There is a separate script "main_fine_tune_mlp.py" for running MLP in which embedding layer is trainable to further fine-tune the embeddings to the current task.

Here are some examples of running "main.py" and "main_fine_tune_mlp.py":

- Multilingual Training on English and Testing on all languages: python main.py –mode="multi" –model-choice="mlp" –multi-train="en" –multi-model-file="multiCCA_512_normalized" –embed-dim=512
- Multilingual Training and Testing on all languages using multilingual embeddings and fine-tuned MLP: python main_fine_tune_mlp.py –mode="multi" –model-choice="mlptune" –multi-train= "en,de,fr,it" –multi-model-file="multiSkip_40_normalized" –embed-dim= 40 –epochs=100
- Monolingual Training and Testing on English using CNN: python main.py –mode="mono" –model-choice="cnn" –language= "English" –embed-dim=300
- Monolingual Training and Testing on English using fine-tuned MLP: python main_fine_tune_mlp.py –mode="mono" –model-choice="mlptune" –language="English" –embed-dim=300
- Monolingual Training and Testing on Italian using MLP: python main.py –mode="mono" –model-choice="mlp" –language="Italian" –embed-dim=300

To get the whole list and description of options with which the scripts can be executed, have a look at get_args.py script. The main flags to provide/change depend on the type of the experiment to be executed:

- Monolingual/Multilingual model: mode="mono" / "multi"
  - If monolingual model is chosen, the flag language need to be specified. The currently supported languages are "English", "French", "Italian", and "german". In this case, the program will only focus on that particular language by training, validating and testing on it. There is a default monolingual embedding model path for each language. If you want to try another gensim model, then go to –w2v-en, w2v-de, w2v-fr or w2v-it.
  - If multilingual model is chosen, flag multi-train need to be specified for example:
    - en: mean training and validating on English only and testing on all languages (English, german, Italian, French)
    - fr: mean training and validating on French only and testing on all languages (English, german, Italian, French)
    - de: mean training and validating on german only and testing on all languages (English, german, Italian, French)
    - it: mean training and validating on Italian only and testing on all languages (English, german, Italian, French)
    - en, de: mean training and validating on English and german and testing on all languages (English, german, Italian, French)
    - en,de,fr,it: mean training, validating and testing on all four languages
- In case of multilingual embeddings, you can specify the directory and model name in –model-dir and –multi-model-file respectively. Don't forget to change –embed-dim accordingly.
- Choice of Document Classification Model: model-choice="mlp" or "cnn" or "svm"
- Choice of the dataset: it is by default rcv (Reuters) dataset. If you provide the raw dataset in the form of folders of xml files for each language, it will parse and preprocess it from scratch. Otherwise, if you the preprocessed version of x_train, x_dev, and x_test (where each split is a list of document saved in the form of pickle where each document element is a list of sentences in the document), you can skip the parsing and provide any dataset you want non-tokenized and non-indexed. You can further adapt it by skipping some steps from the pipeline and providing the preprocessed version directly.

- Preparing the embeddings model: The embedding model should be a UTF-8 encoded plain text file where each line is a word. Each line begin with a lowercased surface form, prefixed by the 2-letter ISO 639-1 code of the language (e.g., "en:school" or "fr:école") followed by the floating-point values of each dimensions in the word embedding. All fields must be delimited with one space, and each line must end with a "" (as opposed to " ° ". If the format of your embedding is different feel free to adapt function: load_multi_vectors in data_utils.

For Launching experiments with different training modes using a particular text classification architecture, you can use scripts in BashScripts. The analysis of the results is done separately using Jupyter Notebooks for better visualizations and enable easier debugging and plot generation. Please have a look at New Experiment Results.ipynb for the generation mechanism of the latest results.

C.0.2.3 [CLDCEval_FineGrained]: Fined Grained Cross-Lingual Document Classification:

The same code and methodology in CLDCEval is extended to include also Fine-Grained Multi-Label Dataset. In other words, this is a generalization of CLDCEval that works for fine-grained and coarse-grained datasets.

C.0.2.4 [ChurnDetEval]: Cross-Lingual Churn Detection:

This contains code to benchmarking different multilingual embedding model as they are applied to churn detection across two languages: English and German. Code is taken and adapted from Maxime Coriou and Athanasios Giannakopoulos from their work: "Everything Matters: A Robust Ensemble Architecture for End-to-End Text Classification".

There are two ways to evaluate Churn Detection either with or without Cross Validation. In the former case, running script run_model_cross_val.py, which expects one Tweet file and bot conversation file per language where each line represents a tweet or bot utterance delimited by its churn label, will go over the whole end-to-end pipeline: preprocess data (CrossValDataset.py), create 10 fold cross-validation which creates 10 different training and testing splits (without validation), train models (NeuralNets.py with metrics defined in metrics_no_dev.py).

To change the default parameters used for running the scripts, adapt get_args.py as needed:

- –word-embeddings-path: path to the monolingual or multilingual embeddings
- –train-mode: the languages (separated by comma) on which the model is trained: en or de for monolingual training or en,de for multilingual training
- –dataset= the choice of the dataset (default is churn)
- –network-type and the hyperparameters related to it

Bash Scripts ready for running example experiments are also provided. We analyze the results using Jupyter Notebook: "Analyzing Cross Validation Churn Detection Results.ipynb"

C.0.2.5 [EvDetEval]: Cross-Lingual Event Detection

This is a library for training unsupervised multilingual event detection. This source code consists of four modules:

- WorldCupTwitterCollection: This is a streaming application using tweepy. The main script to run here is stream_tweets.py which is collects tweets based on hashtags. We create a Twitter application for each language which can be specified in –lang argument. Consumer key, consumer secret, access token and access token secret should be provided in the configuration file stream_config.yaml. There are different hashtags files: officialTags, teamTags1, teamTags2. The output for each hour of each particular day of collection is saved as a separate json file.
- PreprocessingModule: There are three different scripts depending on the type of preprocessor: SpaCy (preprocess_sp_tweet.py), TreeTagger(preprocess_tt_tweets.py) or tokenization only(preprocess_tweet.py) which depends on the language. Prior to that, it is necessary to prepare the file by processing json hour files and converting them into one file for tweets in a particular day and language. This is done using prepare_csv_data.py

- BurstySegmentExtraction: This module defines functionalities for splitting tweets into time windows and sub-windows, analysis of burstiness of unigrams in the tweets for each language alone and for the aggregation of many languages. Run detect_bursty_segments.py to get a feel of how it works where the input is the path to the Twitter dataset, the set of languages, start and end time of the time window and the output is the list of terms with their burstiness scores.
- EventSegmentClustering: this module defines functions for converting bursty triggers to their embedding representation, computing semantic similarity using either embeddings or tf-idf and clustering using knn.
- PostFiltering: This module applies only to multilingual extension choice 2 (clustering all triggers then selecting bursty event clusters). It computes burstiness of events based on the burstiness scores of its constituing triggers.

By running main.py, you can train an end to end system which executes all modules in the pipeline and gives back event clusters for different match sets. The inputs to the script are the following arguments:

- –data-choice: path to the Twitter dataset (world cup 18 by default)
- –round-choice: which part of the world cup to analyze. This can be either Group Round (1st), Group Round (2nd), Group Round(3rd), Round of 8, Quart-Finals, Semi-Finals, 3rd Place, Final. A fixture with metadata (start time and day, teams involved, round, times of goals, cards, etc) needs to be provided for each match in one csv file.
- –trigger-mode: "mono" for multilingual trigger burstiness or "multi" for multilingual event burstiness (options 1 or 2 in multilingual extension respectively)
- –sem-sim-mode: either word embeddings "emb" or "tf-idf".
- –event-mode: either "mono" to perform event detection for each language independently among the languages of the involved teams in addition to English or "multi" for the aggregation of all languages using multilingual embeddings.
- –time-window-size: default is 2 hours which is the duration of the game
- –sub-window-size: default is 1 hour
- –mono-model-dir, –multi-model-dir, –multi-model-file for directories and name of monolingual and multilingual embeddings files
- –neighbor, –min-cluster-segments: number of minimum number of nearest neighbor and minimum number of triggers per clusters in the clustering algorithm

Bibliography

Christian Abbet, Meryem M'hamdi, Athanasios Giannakopoulos, Robert West, Andreea Hossmann, Michael Baeriswyl, and Claudiu Musat. Churn intent detection in multilingual chatbot conversations and social media. In *Association for Computational Linguistics*, 2018.

Hamed Abdelhaq, Christian Sengstock, and Michael Gertz. Eventweet: Online localized event detection from twitter. *Proc. VLDB Endow.*, 6(12):1326–1329, August 2013. ISSN 2150-8097. doi: 10.14778/2536274.2536307. URL `http://dx.doi.org/10.14778/2536274.2536307`.

Hadi Amiri and Hal Daume. Target-dependent churn classification in microblogs. *AAAI*, pages 2361–2367, 2015.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A. Smith. Massively multilingual word embeddings. *CoRR*, abs/1602.01925, 2016. URL `http://arxiv.org/abs/1602.01925`.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL `http://arxiv.org/abs/1409.0473`.

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

Cody Buntain. Language-agnostic event detection across sports from twitter using temporal features. In *Workshop on Large-Scale Sports Analytics*, KDD LSSA '14. ACM, 2014.

Sarath A. P. Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh Khapra, Balaraman Ravindran, Vikas C Raykar, and Amrita Saha. An autoencoder approach to learning bilingual word representations. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1853–1861. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5270-an-autoencoder-approach-to-learning-bilingual-word-representations.pdf`.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. URL `http://arxiv.org/abs/1412.3555`.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November 2011. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1953048.2078186`.

Alexis Conneau, Guillaume Lample, MarcÁurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *CoRR*, abs/1710.04087, 2017. URL `http://arxiv.org/abs/1710.04087`.

Hakan Demir and Arzucan Özgür. Improving named entity recognition for morphologically rich languages using word embeddings. In *Proceedings of the 2014 13th International Conference on Machine Learning and Applications*, ICMLA '14, pages 117–122, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-7415-3. doi: 10.1109/ICMLA.2014.24. URL `http://dx.doi.org/10.1109/ICMLA.2014.24`.

Ali M. Ertugrul, Burak Velioglu, and Pinar Karagov. Word embedding based event detection on social media. In *Hybrid Artificial Intelligent Systems: 12th International Conference*, pages 383–388, La Rioja, Spain, November 2017. Asian Federation of Natural Language Processing. URL `http://www.aclweb.org/anthology/I17-2065`.

Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. In *EACL*, pages 462–471. The Association for Computer Linguistics, 2014.

Daniel C. Ferreira, André F. T. Martins, and Mariana S. C. Almeida. Jointly learning to embed and predict with multiple languages. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2019–2028. Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-1190. URL `http://www.aclweb.org/anthology/P16-1190`.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 748–756, Lille, France, 07–09 Jul 2015. PMLR. URL `http://proceedings.mlr.press/v37/gouws15.html`.

Mourad Gridach, Hatem Haddad, and Hala Mulki. Churn identification in microblogs using convolutional neural networks with structured logical knowledge. *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 21–30, 2017.

Zellig Sabbatai Harris. Distributional structure. *Word. Journal of the linguistic circle of New York*, 10:2–3,146–162, 1954.

Karl Moritz Hermann and Phil Blunsom. Multilingual Distributed Representations without Word Alignment. In *Proceedings of ICLR*, April 2014. URL `http://arxiv.org/abs/1312.6173`.

Zainal M. Khursiah and Mohamad S. Junita. An oil fraction neural sensor developed using electrical capacitance tomography sensor data. *Sensors*, 13(9):11385–11406, 2013. ISSN 1424-8220. doi: 10.3390/s130911385. URL `http://www.mdpi.com/1424-8220/13/9/11385`.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751, 2014. URL `http://aclweb.org/anthology/D/D14/D14-1181.pdf`.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. Inducing crosslingual distributed representations of words. In *Proceedings of COLING 2012: Technical Papers*, pages 1459–1474, Mumbai, India, December 2012.

Philipp Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT, AAMT. URL `http://mt-archive.info/MTS-2005-Koehn.pdf`.

David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397, December 2004. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1005332.1005345`.

Chenliang Li, Aixin Sun, and Anwitaman Datta. Twevent: Segment-based event detection from tweets. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 155–164, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1156-4. doi: 10.1145/2396761.2396785. URL `http://doi.acm.org/10.1145/2396761.2396785`.

Thang Luong, Hieu Pham, and Christopher D. Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, 2015.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 142–150, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. ISBN 978-1-932432-87-9. URL `http://dl.acm.org/citation.cfm?id=2002472.2002491`.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013b. URL `http://arxiv.org/abs/1310.4546`.

Nikola Mrksic, Ivan Vulic, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gasic, Anna Korhonen, and Steve J. Young. Semantic specialisation of distributional word vector spaces using monolingual and cross-lingual constraints. *CoRR*, abs/1706.00374, 2017. URL `http://arxiv.org/abs/1706.00374`.

Thien Huu Nguyen and Ralph Grishman. Event detection and domain adaptation with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 365–371. Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-2060. URL `http://www.aclweb.org/anthology/P15-2060`.

Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *CoRR*, abs/1702.03859, 2017. URL `http://arxiv.org/abs/1702.03859`.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*, 2013.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. Cross-lingual models of word embeddings: An empirical comparison. *CoRR*, abs/1604.00425, 2016. URL `http://arxiv.org/abs/1604.00425`.

George Valkanas and Dimitrios Gunopulos. How the live web feels about events. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 639–648, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2263-8. doi: 10.1145/2505515.2505572. URL `http://doi.acm.org/10.1145/2505515.2505572`.

Dingquan Wang, Nanyun Peng, and Kevin Duh. A multi-task learning approach to adapting bilingual word embeddings for cross-lingual named entity recognition. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 383–388. Asian Federation of Natural Language Processing, November 2017. URL `http://www.aclweb.org/anthology/I17-2065`.

Zhibo Wang, Long Ma, and Yanqing Zhang. A novel method for document summarization using word2vec. *2016 IEEE 15th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC)*, pages 523–529, 2016.

Zhao Yan, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. Docchat: An information retrieval approach for chatbot engines using unstructured documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 516–525. Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-1049. URL `http://www.aclweb.org/anthology/P16-1049`.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, 2016.

Huiwei Zhou, Long Chen, Fulin Shi, and Degen Huang. Learning bilingual sentiment word embeddings for cross-language sentiment classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Conference on Natural Language Processing*, pages 26–31, Beijing, China, July 2015.