Universiteit Leiden

Master Computer Science

Broad Language Support for Automatic Translation
Insight Extractors of Complex Language Patterns

Name:	Olzhas Aldabergenov
Student ID:	s1928643
Date:	27/07/2021
Specialisation: :	Computer Science and Science-Based Business
1st supervisor:	Dr. Bas van Stein
2nd supervisor:	Dr. Suzan Verberne

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Machine learning algorithms can now handle a wide range of natural language processing (NLP) tasks in order to extract meaningful information. The ability to obtain valuable information, on the other hand, is heavily reliant on the proper representation of language properties. Many techniques have been developed to convert lexical units such as sentences and words into numerical form while retaining their meaning. One of the most recent and practical approaches is language-independent or multilingual representations, which can handle multiple languages with a single technique. We conducted experiments with existing approaches such as LASER and LaBSE in this study and analyzed their performance to determine whether they are truly language-independent. In addition, we investigated the effect of language characteristics on the performance of machine learning models. ZyLAB, an eDiscovery software company, employs machine learning models to detect complex verbal patterns. These classifiers are trained on annotated data sets, one for each language. Language-independent data sets are critical for dealing with multilingual NLP tasks.

Contents

1 Introduction **6**
- 1.1 Problem statement . . . 7
- 1.2 Thesis contributions and scope . . . 7
- 1.3 Research question . . . 8
- 1.4 Thesis structure . . . 8

2 Background and related work **9**
- 2.1 Word Embeddings . . . 9
  - 2.1.1 Bag of Words and TF-IDF . . . 9
  - 2.1.2 Word2Vec and GloVe . . . 10
  - 2.1.3 FastText . . . 11
  - 2.1.4 Transformers . . . 12
- 2.2 Multilingual natural language processing . . . 14
  - 2.2.1 Machine translation . . . 14
  - 2.2.2 Cross-lingual embeddings . . . 14
  - 2.2.3 Sequence-to-sequence encoder and decoder . . . 15
- 2.3 Language properties . . . 16
  - 2.3.1 Multilingual properties . . . 17
- 2.4 Multilingual NLP tasks . . . 17
  - 2.4.1 Named entity recognition . . . 18
  - 2.4.2 Sentiment analysis . . . 18
  - 2.4.3 Emotion classification . . . 19

3 Proposed pipeline **21**
- 3.1 Overall Architecture . . . 21
- 3.2 Dataset . . . 22
  - 3.2.1 SST5 . . . 23
  - 3.2.2 SemEval 2018 . . . 24
- 3.3 Machine translation . . . 25
- 3.4 Text preprocessing . . . 26
  - 3.4.1 Basic cleaning operations . . . 26
  - 3.4.2 Text tokenization and normalization . . . 26
- 3.5 Representation learning . . . 27
  - 3.5.1 Langauge Agnostic Sentence Representation . . . 27
  - 3.5.2 Language Agnostic BERT Sentence Embedding . . . 29
- 3.6 Text classifier . . . 29
  - 3.6.1 Support Vector Machines (SVM) . . . 29
    - 3.6.1.1 Parameter tuning . . . 30
    - 3.6.1.2 Kernel . . . 31
    - 3.6.1.3 Multiclass classification . . . 31
- 3.7 Evaluation . . . 31
  - 3.7.1 Micro Averaged F1 Score . . . 32

4 Results **33**
4.1 Baseline . . . 33
4.1.1 FastText results . . . 34
4.2 LASER . . . 37
4.2.1 Similarity analysis . . . 37
4.2.2 LASER results . . . 39
4.3 LASER Family Group (LASER-FG) . . . 41
4.4 LaBSE . . . 44

5 Conclusion and Discussion **45**

6 Future work **49**

7 References **50**

List of Tables

1 SST5's five randomly chosen sentences with their labels 24
2 New distribution of SST5 dataset 24
3 Five randomly chosen sentences with their labels from SemEval 2018 task 1EC dataset 25
4 Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using FastText word embeddings. 34
5 Micro averaged F1 scores on the SemEval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained SemEval 2018 task 1EC training dataset by using FastText word embedding. 35
6 SST5 all language pairs similarity errors in percentage (langs - languages, de-German, nl-Dutch, sv-Swedish, en-Englis, es-Spanish, fr-French, it-Italian, pt-Portuguese, pl-Polish, ru-Russia, cs-Czech, sk-Slovak). 36
7 SemEval 2018 task 1EC all language pairs similarity errors in percentage (langs - languages, de-German, nl-Dutch, sv-Swedish, en-Englis, es-Spanish, fr-French, it-Italian, pt-Portuguese, pl-Polish, ru-Russia, cs-Czech, sk-Slovak). 37
8 Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using LASER embeddings. . 39
9 Micro averaged F1 scores on the Semeval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained Semeval 2018 task 1EC training dataset by using LASER embeddings. 40
10 The results of LASER-FG compare to the SST5 baseline and LASER in percentage. 41
11 The results of LASER-FG compare to the SemEval 2018 task 1EC baseline and LASER in percentage. 41
12 Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 family groups training dataset using LASER embedding. 42
13 Micro averaged F1 scores on the SemEval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained emEval 2018 task 1EC family groups training dataset using LASER embedding. 43
14 Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using LaBSE embedding. . . 44
15 Averaged F1 scores of FastText, LASER, LASER-FG and LaBSE results. . . . 46

1 Introduction

Natural Language Processing (NLP) has seen rapid improvements in recent years, resulting in significant performance gains on a broad range of downstream NLP tasks. A large part of this success can be attributed to the development of large-scale pretraining methods for word representations [1], [2]. Furthermore, there has been a rise of research info continuous vector representations of sentences [3], [4]. In most cases, word embeddings are obtained through the use of a Recurrent Neural Network (RNN) encoder, which is typically trained in an unsupervised manner over extensive collections of unlabeled corpora. Then, Natural Language Inference (NLI) data could result in more competitive results by training the encoder [5]. In later years, it was eventually extended to multitask learning, which included integrating several training objectives such as skip-thought training, NLI training, and machine translation training [6], [7]. These strategies generate priors that are informed by linguistic information, which allows for fine-tuning to more task-specific word and sentence representations to be produced.

Nevertheless, these pretrained representations are monolingual, and the pretraining procedures necessitate a large quantity of training data to be implemented successfully. As a result, many of these models and the success they bring to NLP are in reality largely restricted to a high-resource languages [8]. Recent studies have focused on the construction of models with more extensive cross-lingual applicability, bringing a new surge to the field of multilingual NLP. So far, research in this area has resulted in the creation of multilingual word embeddings [9], where most of the studies focus on cross-lingual word embeddings [10]. As an increasingly common alternative, using word embeddings which share knowledge of multiple languages in a common space [11] or even without supervision [12].

A more competitive strategy uses sequence-to-sequence encoder-decoder architecture [13]. Encoder-decoder is trained end-to-end using parallel corpora comparable to multilingual neural machine translation [14]. This decoder is then destroyed, while the encoder is preserving embed sentences in any training language. Finally, the transformer model was one of the innovations that led to significant advancements in NLP [15], [16]. It is accomplished by using a multi-head self-attention mechanism and various positional encodings to signal token order. As a result, models such as Language Agnostic Sentence Representation (LASER)[17] and Language Agnostic BERT Sentence Embedding (LaBSE)[18] were produced.

These multilingual models are collaboratively trained to execute NLP tasks for several languages to project semantically related words and phrases from different languages into a common multilingual semantic space. Because of this, they provide encodings in which words and phrases with similar meanings receive comparable representations, regardless of the language from which they originate. As a result, they strive to capture semantic meaning more universally. However, even though this work has been successful in that it has enabled effective model transfer across a wide range of languages [19], significant effort is still being put into strengthening the language-agnosticism of multilingual encoders even further. Some studies suggest that multilingual NLP systems benefit from typological knowledge [20], [21]. As a result, it can benefit from improvement in a task such as POS tagging and named entity recognition (NER).

1.1 Problem statement

ZyLAB is an eDiscovery[22] software company that uses artificial intelligence and data science to automate mundane processes and tedious tasks and enable data-driven decision-making in the legal industry. It has developed a number of binary extractors to detect complex verbal patterns such as attitudes, emotions, profanity, intimidation, pressure, and other similar patterns of expression. These binary classifiers make use of a variety of machine learning techniques that are trained on annotated data sets to achieve their classification accuracy. It is necessary to construct a separate data set for each language. Such data sets are created using various hybrid manual-automatic processes in ZyLAB, utilizing tools such as Snorkel[23] and machine translation[24], among other things.

Recent experiments with language-independent representations have revealed that, under certain conditions, for instance, teaching an English classifier with a German annotated dataset can also improve the quality of the model, but French and Spanish, according to the results of the experiments, decrease performance [25]. Additionally, training French classifiers with language-independent English and Italian data improves performance, whereas training French classifiers with German or Dutch data causes the systems to become confused and malfunction. It has been demonstrated that additional preprocessing and postprocessing can alleviate these issues. Clearly, specific linguistic properties of language families such as concatenate words, phonetic and other parameters have an impact on the classifier behavior.

1.2 Thesis contributions and scope

We want to study as many as possible languages from the beginning of our study, and we select 37 of them. Here is a list of these languages: Basque, Bulgarian, Catalan, Croatian, Czech, Danish, Dutch, English, Estonian, Finnish, French, Galician, German, Greek, Hungarian, Irish, Italian, Latvian, Lithuanian, Maltese, Polish, Portuguese, Romanian, Slovak, Slovene, Spanish, Swedish, Russian, Turkish, Arabic, Farsi, Chinese, Korean, Japanese, Hindi, Urdu, and Tamil.

The main point of this thesis is to understand the relative merits of several language-independent representations. Before commencing, we wanted to set our performance benchmarks for each NLP task to use for our investigation. For us, the best approach was the FastText word embeddings. It utilizes a single pre-trained model for each language, and there is no shared knowledge between the languages. The second main topic of research was to choose required datasets as we consider complex NLP tasks such as sentiment analysis and emotion classification. We chose SemEval 2017 task 4a [26], SemEval 2018 task 1EC[27], and Stanford Sentiment Treebank (SST5)[28] for this goal. Those datasets have different contexts; for instance, SemEval datasets collection of tweets and SST5 is movie reviews. We translated all these datasets to 37 languages using Google Translator's cloud API as original datasets have covered several languages. Afterward, we trained the SVM classifier with all listed datasets of 37 languages and evaluated the model using a running test. Then, we decide to scope the number of languages to 12 as it was hard to handle the training classifier with different combinations. The twelve languages are: Enlgish, Dutch, German, Swedish, Russian, Slovak, Polish, Czech, Spanish, French, Italian and Portuguese.

1.3 Research question

Specifically, we are interested in conducting an experiment employing existing language-independent methodologies and analyzing their performance in order to determine whether they are actually language-independent or not in this study. For this purpose, we selected two methods as LASER[17] and LaBSE[18]. Then, we used their pre-trained models and considered them feature extractors. At the same time, we would like to investigate the impact of language attributes on the classifier's performance. By answering following research questions of this thesis:

Q1 How can language-independent word representations be used for more complex linguistic tasks such as sentiment analysis and emotion classification?

Q2 What are the results of the classifier trained with language-independent representations for selected twelve languages?

Q3 Do the linguistic properties have an effect on the classifier? In case it does:

- **Q3.1** What type of linguistic properties is responsible for specific errors?
- **Q3.2** Is it possible to address classifier errors by using pre-processing or post-processing methods?

1.4 Thesis structure

The following is the order in which we completed our thesis. First and foremost, in Chapter 2, we provided background information on the variety of related works. Then, in Chapter 3, we went over our proposed approach in greater detail. The results of the experiments that were carried out are presented in Chapter 4. Finally, in Chapter 5, we concluded and offered some suggestions for feature work in Chapter 6.

2 Background and related work

Nowadays, machine learning algorithms can handle a wide range of natural language processing (NLP) tasks, such as extracting meaningful information from text[29], [30]. This extracted data may take the form of objects, relations, events, or entities. Obtaining valuable information, on the other hand, is highly dependent on the proper representation of language properties. Many techniques had been developed to convert language units like sentences and words into numerical form while retaining lexical meaning [1], [2]. Language-independent or multilingual representations, which can handle multiple languages with a single technique, are one of the most recent and practical approaches [17], [18]. Working with models that can transfer knowledge to as many languages as possible is critical in today's globalized world. Because most NLP algorithms rely on the supervised algorithm, which is highly dependent on the annotated dataset, low-resource languages can benefit significantly from language-independent models. In this study, we want to run experiments with existing approaches and analyze their performance to see if they are truly language-independent. Furthermore, we would like to investigate the impact of language properties on the performance of machine learning models. This chapter will provide an overview of the evolution of word and sentence representations, as well as how they are used in various NLP tasks.

2.1 Word Embeddings

As we mentioned before terms (words) and expressions can be constituted with word representations in the form of word embeddings. In the early stage, Clark and Pulman [31] adapted a method from cognitive science to combine the symbolic and distributional approach to represent words using tensor products. Mitchell and Lapata [32] proposed a framework using additive and multiplicative functions to represent phrases and sentences in vector space. Mikolov and Sutskever[1] presented different extensions to improve the training speed and the vectors' quality by using subsampling of frequent words. Their study showed that learning good vector representation for millions of phrases is possible by presenting a simple algorithm for finding expression in the text. Then, Pennington [2] offered a global log bilinear regression model, which efficiently takes advantage of mathematical information by training non-zero elements in a word-to-word matrix and generating a meaningful dimensional space out-performing the related models in entity recognition. This section will give detailed information regarding embeddings, which are primarily used in practice.

2.1.1 Bag of Words and TF-IDF

One of the widely-used models used for representing words is called the bag of words (BoW). It refers to a method of representing text with a focus on the frequency of words in any given document. Listing each unique word starts with the first word to appear in the document thus does not consider the order in which words occur [33]. Word count determines the frequency of occurrence of each word. However, it leads to the sparse matrix problem, especially when there are many new words represented as 0s, and at the same time, vectorial representation becomes longer. Term Frequency-Inverse Document Frequency (TF-IDF) is an advanced version of the BoW. It is focusing on the importance of any particular word and its frequency of occurrence. The TF-IDF is dot product Term Frequency (TF) and the Inverse Document Frequency (IDF) as shown following equation:

$$tfidf(t, d, D) = tf(t, d) \times idf(t, D)$$

Where the *t* refers to the number of the word we counting in documents *d* inside document collections of *D*. Term frequency (TF) involves calculating the frequency of any given word [34].

$$tf(t, d) = \frac{n_{t,d}}{\sum_k n_{k,d}}$$

In a turn of inverse document frequency (IDF), it is determining the importance of a word by calculating the IDF score or value. Rarely occurring words have higher values than those occurring more frequently.

$$idf(t, D) = log \frac{|D|}{|d \in D : t \in d|}$$

TF-IDF also has certain limitations. For instance, the technique fails to identify similar words when minor changes such as tense or plural are made [34]. In addition, it fails to compare different words or show simultaneously occurring words. Overall, both BoW andTF-IDF do not consider contexts such as semantics, grammatical structure, or order of words [34], [35].

2.1.2 Word2Vec and GloVe

Word2Vec is the next level of embeddings that consider the context of words where TF-IDF and BoW have a problem. It can utilize unsupervised training to help encode words into useful dimensional vectors [36]. Word2Vec representation model derives the meanings of words and sentences using the skip-gram model [37]. It is more effective in providing representations for limited data and also for less used words. Word2Vec also uses CBOW(Continuous Bag of Words), which is beneficial. It is quicker to use and more effective than statistical language modeling in providing representations of words that occur more often.
An exciting alternative to Word2Vec is GloVe embeddings which consider whether two or more words co-occur using global data. It uses global matrix factorization methods and local context window methods. An example of models used in the former is the latent semantic analysis (LSA). An example of models used in the latter is the skip-gram model [2]. The use of these representation models presents particular challenges. LSA effectively analyzes statistical data but fails to provide a practical comparison or similarity analysis for words [2]. Additionally, the use of the skip-gram model is effective in providing comparison or similarity of words. However, the skim-gram model does not provide a good analysis of the statistical data since it considers local context rather than the global data in providing word analogy.

Both Word2Vec and GloVe models also present another problem. Word analogy only considers specific contexts [38], [39]. It makes the models ineffective as words are used in diverse contexts. It is also unreliable when providing an analogy for words with different meanings but coming from or used in the same place. Addressing these challenges involved making different representations for different meanings of a word [39]. Also, researchers considered deriving subwords to help provide better word analogy [38]. These challenges impeded further research in distributional representations.

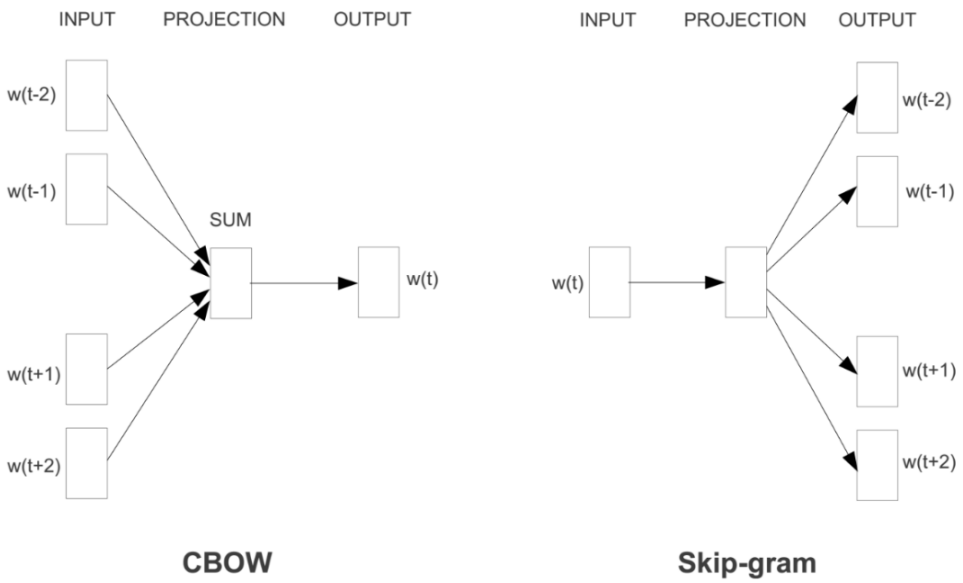

Figure 1: Word2Vec architecture: CBOW and Skip-Gram

2.1.3 FastText

For technical reasons, we did not use the previously mentioned vector representations and word embeddings in our study. Along with the limitations we have already discussed, it has one additional significant limitation that hinders the progress of our investigation. The fact that they are all language-dependent means that they cannot be considered in our study as we looking for the opposite. Fortunately, FastText was able to resolve the issue.

FastText is an extension of the Word2Vec model. It is a representation method that uses vectors developed by the artificial intelligence (AI) team of Facebook. It helps to ensure effective and quick execution of tasks it is designed to do. While Word2Vec is more effective in generating semantics, FastText has a different role. FastText is designed to learn how words are composed or their syntax by analyzing the morphological data of the text. Compared to Word2Vec and Glove, FastText is more effective as it can do tasks using new or rare words, while Word2Vec and Glove only work with words included in their dictionaries. FastText does this by subdividing the new or rare word into n-grams. It is then used to generate word embeddings.

According to the authors [40], the model is a simple neural network with only one layer. The BoW representation of the text is first filled into a lookup layer, where the embeddings for each word are fetched. The word embeddings are then averaged to create a single averaged embedding for the entire text. We end up with n x d number of parameters at the hidden layer, where d is the size of the embeddings and n is the words in the vocabulary size. Then, a single vector is left after averaging, which is then fed to a linear classifier. We apply the softmax over a linear transformation of the input layer's output. The final log-likelihood function shown below:

$$-\frac{1}{N}\sum_{n=1}^{N} y_n \log(f(BAx_n)$$

Where the f is the softmax function, A is look up matrix, B is the linear transformation, and x_n is the one-hot-encoded word representation.

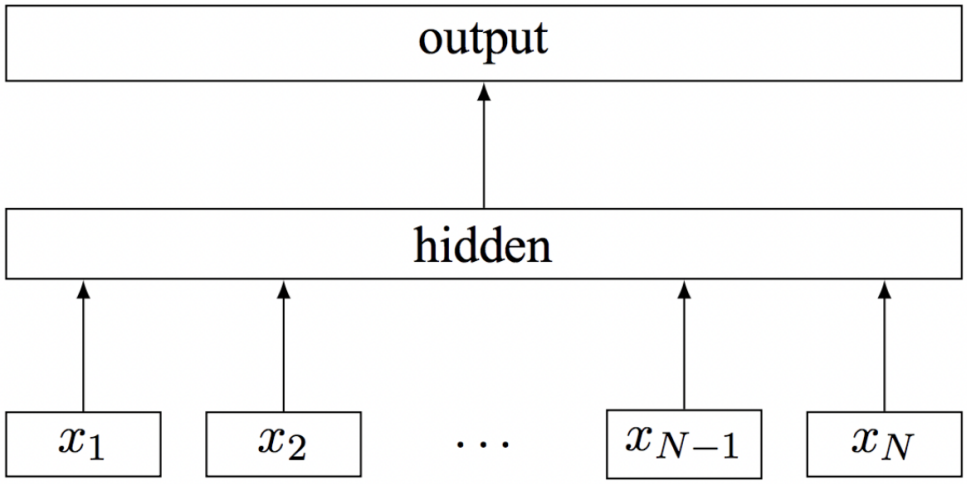

Figure 2: Model architecture of FastText

FastText helps to enhance accurate text classification. Text classification requires taking specific pre-processing steps such as cleaning text from unnecessary items and conducting a tokenization process that increases the risk of inaccuracy in text classification [41]. However, FastText is effective than some other algorithms even when the pre-processing steps are excluded [42]. It can help in data classification in different fields, and also can be used for several other purposes. For instance, it helps to efficiently and effectively provide sentiment analysis and handle tasks such as making the most appropriate recommendations for music or videos.

2.1.4 Transformers

The transformer model of neural machine translation was suggested by Vaswani[15] after noting the impact of the attention mechanism. At a high level, this architectural setup could be as efficient as the encoder and decoder model when applied to the sequence-to-sequence tasks. Before its development, the Recurrent Neural Networks and the Long Short-term Memory had been voted the most suitable option for the transduction and modeling of sequence problems since they gather and sequentially incorporate the data. These techniques could also easily manage long-term dependencies with the aid of in-built memory mechanisms [43], [44]. However, the main setback exhibited by the Recurrent Neural Networks is their inability to represent longer sequence representation vectors of consistent lengths efficiently and accurately.
The transformer framework incorporates similar operational principles as the encoder and decoder framework. The encoder enciphers the input sequence representation, followed by the decoder deciphering the intermediary information to generate the output sequence representation. Nonetheless, Vaswani and his team found a means through which the attention mechanism could be used within the encoder, in a unique process referred to as the self-attention mechanism [15]. Thus, similar to the functionality of the attention mechanism, this newly developed approach can efficiently assist in transmitting information from the encoder to the decoder from both ends. Additionally, the self-attention mechanism was found to enhance the

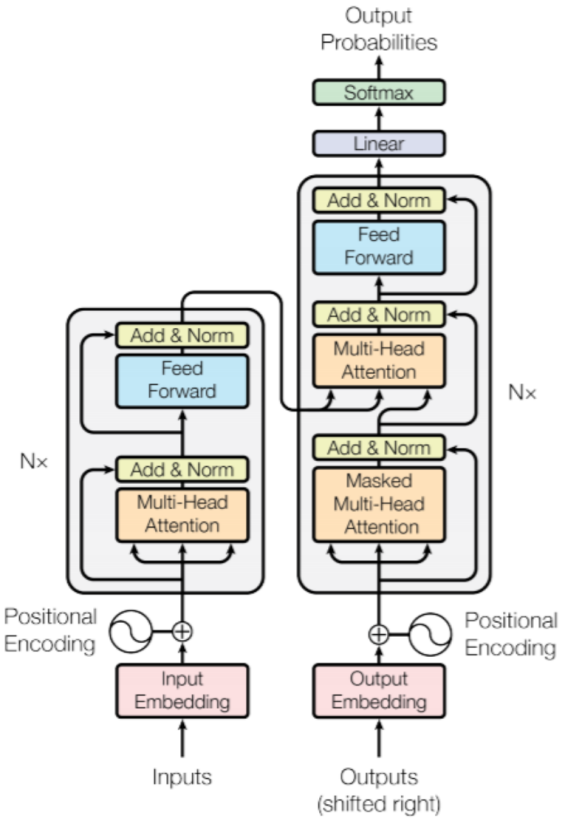

Figure 3: Transformer architecture

effectiveness and efficiency levels of the enciphering processes conducted by the encoder.

The Bidirectional Encoder Representations from Transformers will be evaluated as another decision mechanism in this thesis (BERT). Google's research department introduces it in 2018 as a new model of its kind. It is intended to be a pre-trained deep learning model capable of performing a wide range of different NLP tasks. It is possible because it has been so thoroughly pre-trained that it fully understands how a language is constructed and used [16]. The implementation of the BERT representation is accomplished by using the architecture of a Multi-Layer Bidirectional Transformer Encoder that employs the same configuration as that presented by Vaswani et al.[15]. It is first pre-trained on two unsupervised prediction tasks, namely "Masked Language Modelling" and "Next Sentence Prediction," before being put through its paces. In the case of the former, a percentage of the input tokens is first masked in random before being predicted. The latter examines the relationship that exists between two different sentences. Overall, pre-trained BERT can be used in two different ways: fine-tuning and extracting features. Fine-tuning involves using BERT as the classifier itself while extracting features results in the embeddings being produced as a result. The BERT algorithm is only used as a feature extractor in this study, not as a classifier, and more detailed information about its application can be found in Section 3.5.2.

2.2 Multilingual natural language processing

The success of monolingual representation led to applying the same pretrained methods to the other languages and had attracted significant attention in the development of multilingualism. The multilingual representation uses the extended version of the traditional skip-gram model to predict words from monolingual and cross-lingual languages. In the traditional skip-gram model, monolingual representation neighbors a given word, and cross-lingual neighbors the target word in a parallel sentence and word pair [45]. Pham [46] introduced an approach for distributed embeddings having variable-length text in multiple languages. Their model learned distributed embeddings for phrases and sentences by extending to the bilingual context and efficiently encoding the text sequences of multiple languages. Singla[45] presented an approach to learning multilingual distributed embeddings of text where their system trains the multilingual skip-gram jointly with words and sentence representations. According to Ruder[10], multilingual representations, unlike the other methods involving joint and mapping models, concurrently learn respective representations with the assistance of joint cross-lingual and monolingual functions, as well as parallel train corpora. Many approaches have been developed to handle multilingual tasks. This section explained several of them in detail, including machine translation, cross-lingual embeddings, sentence encoders and decoders, and transformers.

2.2.1 Machine translation

Machine translation formally known as MT, can be used to transfer learning. It is characterized as interpretation from a source language to another target language utilizing automated frameworks and, with or without human assistance [5]. By using MT, the monolingual dataset can be extended to other languages and trained with a classifier. There are different types of machine translation available today. Rule-based machine translation (RBMT) uses linguistic and bilingual rules for the grammatical content of source language into the target language and for every pair set. Statistical machine translation (SMT) works by using a statistical translation algorithm and is translated by using training data with algorithms also selecting commonly observed words from analysis of training data. It learns and compares the training files, and then the source text is translated based on the probability of appearing in the target text. Neural machine translation is another machine translation that works by using a deep learning model.

2.2.2 Cross-lingual embeddings

Cross-lingual embeddings represent words and reason about terms (words) and their meaningfulness in multilingual contexts. So we can say that cross-lingual representations are n-dimensional space embeddings of similar words from multiple languages [10]. Adams[47] studied how to use lexicons to improve language models when there is a total of one thousand sentences in the training dataset. The model learns to cross-lingual word representations for training the monolingual language. Doval[48] applied a transformation after the first alignment step by moving cross-lingual synonyms in the middle. Their purpose was to get a better cross-lingual vector. Lin[49] contributed to the automatic selection of optimal transfer languages and considered it a ranking problem. They also built models to consider the features that are mentioned after for the prediction task. Their model predicted better transfer languages than the extemporary baselines by having a feature separation and providing insights on informative features. The Cruz[50] presented WikiText-TL-39, a new language modeling benchmark in Filipino. Where the language model finetuning techniques like BERT and ULMFiT can consis-

tently train robust classifiers in low-resource settings, with only a 0.0782 increase in validation error when the number of training examples is reduced from 10K to 1K when using a privately held sentiment dataset.

2.2.3 Sequence-to-sequence encoder and decoder

The encoder-decoder architectural model was initially suggested in 2014 by a group of researchers led by Cho and Sutskever [43], [51]. This particular framework supposedly enables a learning process that involves multilingual representations through training objectives using a series transduction problem in the form of machine translation. These two models can be acknowledged as distinct forms of Gated Recurrent Unit networks (GRU) or Recurrent Neural Networks (RNN) [43], [44]. A typical example of RNNs is long-short term memory.

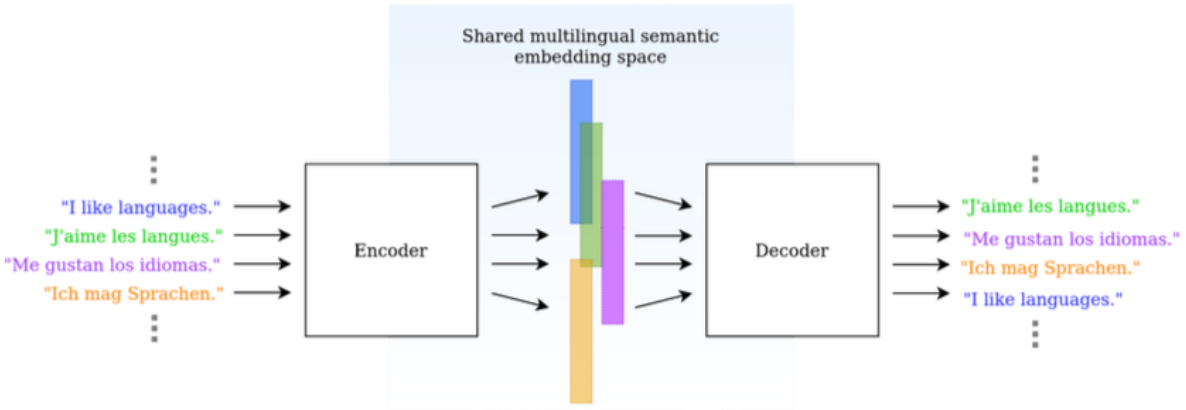

Figure 4: Sequence-to-sequence encoder and decoder architecture

Fundamentally, an input sequence consisting of a source language is fed into the encoder. After receiving this information, the encoder learns to encipher the relative semantics involved, producing an incessant representation vector of a consistent length. The transitional representation is then transmitted to the decoder architectural setup, whose primary function is reconstructing the semantics of the series into a specific language separate from the original one [52]. Simultaneously, the decoder also acts as a source of helpful feedback to the encoder model, especially when the representations do not adequately capture the needed information. Ultimately, both models can be combined with various forms of attention mechanisms for statistical machine translation. The encoder and decoder frameworks were later integrated into the neural machine translation concept in 2014 by Bahdanau and his colleagues [53].

In 2020, a group of researchers conducted a study that analyzed the general performance levels of a specific multilingual translation framework to develop sentences that were consistent in size [54]. The suggested framework used an allotted attention bridge amid encoders and decoders, each based on a specific unconventional language. According to the reports provided, the researchers noted that the classification tasks registered a performance improvement. Synonymously, the translation quality was also enhanced in the high-dimensional sentence representations. On the other hand, the researchers reported that the precision in non-trainable

similarity tasks increased when shorter sentence representations were applied.

The most significant aspect regarding the findings is that encoding in sentence-level linguistic properties can be enhanced through multilingual learning. These conclusive results also align with other similar research findings, which reiterate the premium use of multiple encoder and decoder frameworks in neural machine translation (NMT) to learn joint sentence representations of consistent sizes [13]. In subsequence, the suggested attention bridge layers could be advantageous in retrieving both syntactic and semantic information.

2.3 Language properties

Language is known as an emblematical system, and representation is the illustrative process of any written idea. Linguistic typology can be used to check and identify the similarities and differences between languages. It involves studying the structural composition of different languages across the world and categorizing them accordingly. While there are at least 8 000 languages in the world today, researchers have never agreed on the exact number due to the complex nature of the linguistic typology [55]. One of the reasons is that some different languages or dialects have certain similarities making it possible for people speaking different languages to easily understand each other even when they do not have prior knowledge of the other language [56]. In addition, linguistic typology also involves the study of word order for different languages. The constituents or constituent structure of the language are studied through syntactic analysis. According to Dryer [57], this may involve analyzing the grammatical structure by considering aspects such as how subject, verb, and object (SVO) occur in a sentence. Different languages have different SVO arrangements. There are six main word orders: SVO, SOV, VSO, VOS, OVS, and OSV.

Most languages are spoken in Europe, including English, follow the SVO word order. It means that a grammatically correct sentence should start with the subject, followed by the verb, and then the object. An example of a sentence is 'The baby ate the food'. Here, 'the baby' is the subject, 'ate' is the verb, and 'the food' is the object. In comparison, most languages are spoken in Asia. For instance, Korean and Hindi languages follow a different word order, SOV. In the sentence example provided, SOV order would translate to 'The baby the food ate" in English and as such would be wrong considering the grammatical structure or word order.

Categorization is not limited to the word order only. Researchers also consider studying the structure of different languages by considering the sub-domains of the syntactic constituents. For instance, the structural features in the use of adverbs and noun modifiers such as adjectives, adjunctions, and possessives. These help to categorize language better.

Developing a typological method to categorize languages effectively is impossible. According to O'Horan [58], the language features considered during syntactic analysis cannot be reliably used to categorize languages fully and, as such, should only be used as valuable measures for linguistic typology. While different syntactic structures or structure orders can be used for typological purposes, some are more effective than others. Similarly, there might be a lack of a dominant typological order. In studying and categorizing the French language, the SVO order is more dominant. However, the SOV word order is used if the object in the sentence is a pronoun. Therefore, linguistic typology is a complex field in which many features should be

considered.

2.3.1 Multilingual properties

Typological information may or may not be helpful in multilingual NLP. If the task involving multilingual NLP only focuses on critical concepts without considering language structure, then typological information is not essential. For instance, to retrieve data, the system only considers the keywords entered, and as such only semantic understanding is needed and not typological features [59]. It also applies when researchers need to show universal representations without considering the language structure or typological features of the source language.

In a study done by Cohen [60], typology is proven critical in multilingual NLP. Analyzing the language structure from unannotated sentences of one language is easier and more effective when studied while considering the typological structure of annotated sentences of other languages. Furthermore, studies show that typological information can help to make it easy to effectively do different tasks executed using multilingual NLP systems [20], [21]. As a result, POS tagging and other tasks can be done more effectively.

Different studies discuss the use of data from linguistic typology to execute tasks involved in multilingual NLP. In a study done by Naseem [21], typological data is used to facilitate multilingual dependency parsing. This involved NPL, whereby the syntactic features, particularly word order of the different languages, were considered to ensure that sharing of information was done selectively. Similarly, another study conducted by Zhang [61] also considered typological information in part-of-speech (POS) tagging. Researchers considering using typological information in multilingual dependency parsing postulate that different languages may have some similar syntactic features. They also note that certain syntactic features are only found in some languages. As such, selective sharing possible by using typological information during multilingual dependency parsing can be more effective if the languages have certain typological features in common. In this study, we grouped languages based on their language family group and tried to train the classifier in order to check the performance of the machine learning model.

2.4 Multilingual NLP tasks

Natural language allows for various pragmatic and semantic processes that enable people to communicate in a specific language to express their behavior, feelings, or sentiments. Related processes, such as multilingual representations of words or sentences, allow people to classify the meaning of different words in a diverse context.

Many studies have been conducted on multilingual NLP tasks by involving different languages such as English and German. In order to validate the performance of classifiers, researchers indicate the importance of MLDoc, Reuters, and other corpora in making text classification, particularly cross-lingual classification [25], [62]. It helps text categorization for enormous data resources to identify incorrect information and make appropriate corrections in the model. However, we used a different approach in this thesis. We made a translation of the dataset from one language to other languages and trained the classifier. The motivation behind it is to work on datasets that we are interested in on one side; on another side, we are also inter-

ested in checking how neural machine translation works. As it also considered the language-independent approach. In this section, we made an overview of complex tasks in NLP which can be performed in a multilingual context, including sentiment analysis, emotion classification, and named entity recognition.

2.4.1 Named entity recognition

Named Entity Recognition (NER) is helpful in the extraction of information. Rules help in the detection or recognition of identities and can be developed manually or automatically. Studies indicate that manually developed rules are less applicable for new domains [63]. Different approaches were used, including rule and dictionary-based methodologies which were most common when NER was introduced. The technique involves domain experts, following specific rules, and using multilingual or bilingual approaches [29]. The approach is not portable and may also not include many linguistic data [29], [63], [64]. Second, statistical methods involve developing statistical models by relying on machine learning and using labeled linguistic data from the corpus [29]. Examples include hidden Markov models (HMM) and CFR models. While this model is portable, it relies on data used from the corpus; thus, it is limited in its use [29], [63]. It also requires enormous amounts of data. Lastly, the mixed-methods model is a hybrid of the other two models using the rules of both [29], [63]. It is more advantageous as identifying the target word is easier and faster. Hybrid models help to ensure recognition is based not only on initial names or time but also on recognition based on phrases that may have particular meanings.

2.4.2 Sentiment analysis

Sentiment analysis helps to make emotion classification and also helps in detecting polarities. Research studies show that sentiment analysis focuses on extracting data on two comparable aspects: like or dislike and negative or positive [30], [65]. The technique also involves polarity classification for binary sentiments. As such, this can help in choosing whether identical or different labels should be used. Polarity classification is essential in that it can provide more specific and detailed analysis, for instance, understanding the benefits and drawbacks of a given factor [30], [65]. By classifying polarity or mood, researchers can analyze opinions or sentiments.

Different studies also show that sentiment analysis may present challenges. Poor classification or inaccuracies may occur due to the lack of proper preprocessing of data [66]. Noteworthy, relying on the recognition of emotions using sensors is less effective when compared to sentiment analysis that relies on textual data [66], [67]. Text data is easier to collect and does not require sophisticated equipment to collect. Researchers also note that particular sentiments may not be opinionated but may have a polarity; thus, understanding whether it is subjective or objective is vital to ensure text classification is done effectively [65]. Sufficient data should be collected and appropriately labeled.

Research studies describe the different methods of carrying out sentiment analysis. It includes the machine learning approach, lexical-based approach, and multilingual sentiment analysis [30], [68]. Also, there are different levels of sentiment analysis, including document and sentence level analyses which do not consider the choices or preference of the individual. It may

involve checking whether the sentiment is opinionated or neutral. Aspect level analysis considers the opinions of the person.

Many approaches can be used in sentiment analysis. One of them is a Naive Bayesian classifier that involves determining the probability of classifying a document in any particular class [69]. The model is easier to train and use in classification. The model also determines the polarity of the text. The maximum entropy approach is also used in classification by determining conditional probability distribution. It determines distribution in different classes. This model also analyzes the polarity of the text. A more advanced method is a Support Vector Machines (SVM), which involves finding the decision boundaries while considering the maximum margin possible for any different classes. Compared to the other approaches, SVM enhances precision and accuracy [69]. It helps in classification while also more effectively analyzing margin for any different classes.

2.4.3 Emotion classification

Emotion categorization is vital in emotion classification. Studies conducted by Shao[70] discuss how emotions can be categorized to help multilingual text classification. Researchers use different algorithms to extract texts or speech expressing emotions [71]. The emotion extractor model may also consider the nature or quality of emotions, degree, and intensity [70]. Researchers define six emotions, including anger, sadness, fear, disgust, anxiety, and shame, as negative emotions [68], [70], [72]. Similarly, words such as joy, pride, comfort, affection, satisfaction, and excitement as positive emotions. Other emotions can be considered empathy, calm, surprise, cheerfulness, and enthusiasm [73]. Multilingual text classification thus considers the categorization of different words that define emotions.

Multilingual text classification also involves emotion detection, as described in different studies. Different studies show that each method of emotion classification presents certain drawbacks. First, the lexicon-based method involves tagging inspirational words and collecting statistical data [68]. However, this method is ineffective if a word or phrase is reversed and when a word is not included as a tagged lexicon. Second, rule-based identifications help show or analyze emotions in sentences [70]. However, designing this model is challenging, and it may also lead to ineffectiveness as it only identifies some instances. Lastly, thesaurus-based clustering helps to identify emotions by determining and clustering words that show emotions and have similar meanings [70]. Studies show that some synonyms may be used in different contexts to show emotions or present facts, thus leading to inaccuracies of the model.

Different machine learning approaches are used in emotion classification. First, the Multinomial Naive Bayesian classifier uses the Bayes' theory to classify text by considering the probability of a document being categorized in any given class [74]. The class with the highest probability is calculated. The advantage of this approach is that it is comparatively faster than the other approaches. Second, the Artificial Neural Network refers to classification based on a model that assimilates the neural network of the human brain [74]. It has perceptrons to calculate the value of the input. Three, the K-Nearest Neighbors (KNN) is a technique that identifies similarity metrics to determine K instances that are more common [75]. The model then uses this for classification.

Lastly, the Support Vector Machine (SVM) involves the classification of data using a supervised approach. It involves considering the decision boundary and the ordered sequence of text from different classes. Of the four approaches, SVM is the most effective in classifying emotions as it is more accurate [74], [75]. It properly analyzes the differences between different classes.

3 Proposed pipeline

As previously stated, the primary goal of this thesis is to evaluate language-independent representations in complex NLP tasks. It means that in order to conduct a thorough analysis, our experiment must include as many languages as possible from various families. As a result, we chose 12 languages and divided them into three groups based on their language family group:

- Germanic: English, Dutch, German, Swedish
- Romance: Spanish, Italian, Portugues, and Catalan
- Slavic: Russian, Polish, Slovak, and Czech

The languages in the same group have similarities that they share. On the other hand, they are different comparing with other family groups. We think that this scope gives us enough information to make a conclusion at the end. The second that we focused on in our research is NLP tasks such as sentiment analysis and emotion classification. We decided to exclude doubt that the language-independent representation can also perform differently based on conducted tasks.

3.1 Overall Architecture

Due to the fact that we are not using parallel corpora[1] in our study, it is not easy to evaluate the classifier's performance in each language separately. It is for this reason that we must have a support system in order to complete our research successfully. We decide to translate datasets into other languages as a starting point. After that, train classifiers with different languages and evaluate the performance of classifier performance. English is the source language for both our dataset and Figure 5 illustrates how the English language is used as the source language and then translated into other languages by other languages. It is possible that this approach will have drawbacks because the translation may not be accurate enough, and the classifier may produce only mediocre results. On the contrary, we consider it as a support point, as well as an opportunity to test how well machine-translated data can be used in the machine learning model as a training dataset. In addition, we will have the results of a classifier trained with the English language, which will allow us to compare the results of the source language to those of other translated datasets, which will be helpful at the end.

The overall architecture of our approach is visualized in Figure 6. Many machine learning algorithm pipelines follow the same structure and include all of these steps:

1. Dataset used to train model
2. Text preprocessing
3. Extract features by using embeddings
4. Machine learning model training
5. Evaluation of the machine learning model

[1]Collection of texts that have been translated into one or more languages other than the original.

However, we would like to highlight step 3, which contains two different language-independent embeddings. The distinction is that in our baseline, we use FastText for word representation and sentence representation in our primary experimentation.

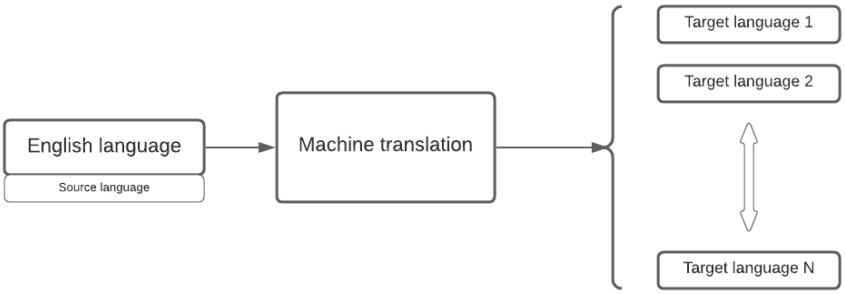

Figure 5: Machine translation

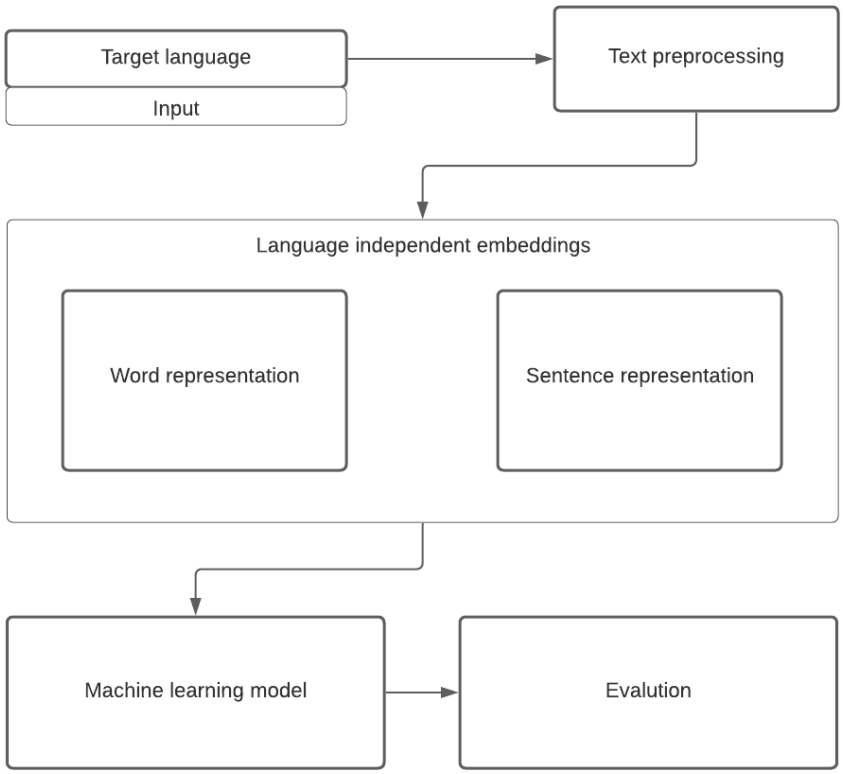

Figure 6: The overall pipeline architecture

3.2 Dataset

Sentiment analysis and emotion classification are two tasks on which we decided to conduct research and experiment. As a result, classifiers required appropriate datasets in order to train the model. We came to the conclusion that the Stanford Sentiment Treebank fine-grained

(SST5)[28] and SemEval 2018 [27] were the best candidates for the tasks we had set for ourselves. Detailed descriptions of each dataset will be provided in this section.

3.2.1 SST5

A popular dataset for working with sentiment analysis is SST5[28], which provides the ability to learn about sentence structure rather than simply observing a single word in isolation. It includes the 11855 movie reviews with 215154 phrases collected from a service called Rotten Tomates[2]. Human annotators labeled each sentence from Amazon Mechanical Turk [3] workers, and they were given a slider with 25 different levels of sentiment. It is critical to remind them that the human perception of text classification has always been the case. It means that the dataset and classification model will never be perfect in a real-world application and thus will always contain some biases. However, as illustrated in Figure 7, we can rely on specific similarities in general. Consider the fact that the shorter phrases are primarily neutral, while the longer phrases have a good distribution of positive and negative classes. The authors were able to do this by creating a fine-grained version of the collected dataset and then segmenting the 25 levels of sentiments into five groups. They reduced the number of sentiment classes from ten to five (very positive, positive, very negative, negative, and neutral). In Table 1, it can be seen how the actual dataset appears, and this is the result.

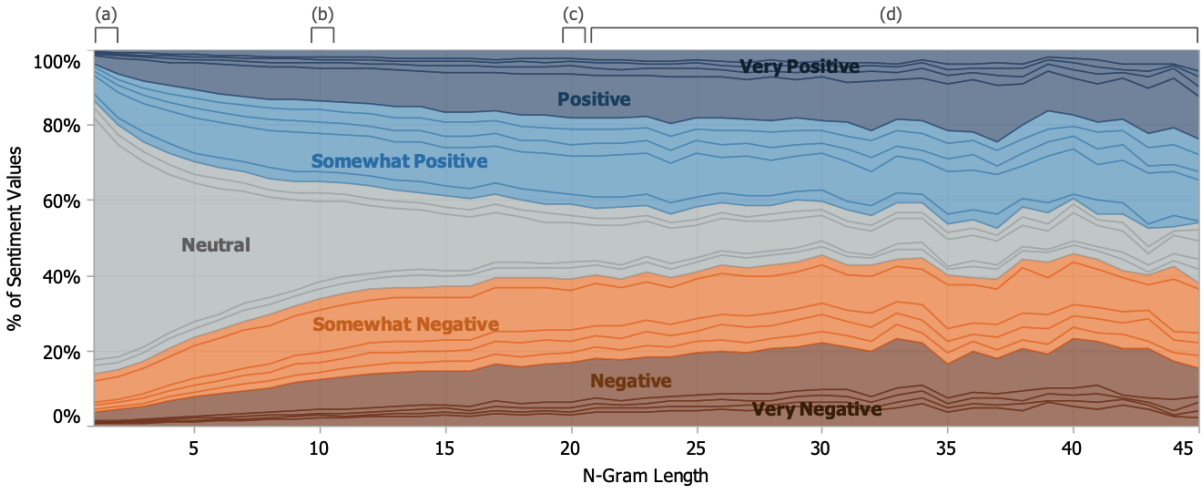

Figure 7: The distribution of SST5 labels

We chose this dataset because it contains only clean human sentiments that have been collected and because the source is the point at which people intend to express their opinions about a particular movie in greater depth. In addition, we want to broaden the scope of this dataset's utility by including data from other languages, as the original dataset only included data from a few languages. The original dataset, on the other hand, is fine-grained, and we have been interested in binary classification from the beginning of our research. Despite the fact that fine-grained classifications are fascinating, they are also known to be more challenging to implement than binary classifications. We want to make the task for our model as simple as possible because we believe it is challenging to convey the same sentiment in different

[2]https://www.rottentomatoes.com/
[3]https://www.mturk.com/

Text	Label
The Rock is destined to be the 21st Century 's new " Conan " and that he 's going to make a splash even greater than Arnold Schwarzenegger , Jean-Claud Van Damme or Steven Segal .	Positive
The gorgeously elaborate continuation of " The Lord of the Rings " trilogy is so huge that a column of words can not adequately describe co-writer/director Peter Jackson 's expanded vision of J.R.R. Tolkien 's Middle-earth .	Very positive
You 'd think by now America would have had enough of plucky British eccentrics with hearts of gold .	o
It 's only in fairy tales that princesses that are married for political reason live happily ever after .	Negative
By no means a slam-dunk and sure to ultimately disappoint the action fans who will be moved to the edge of their seats by the dynamic first act , it still comes off as a touching , transcendent love story .	Negative

Table 1: SST5's five randomly chosen sentences with their labels

languages. We went one step further and decided to create two labels out of five by merging classes such as very positive and positive in one class, then very negative and negative in the same bucket, resulting in two labels out of five. Consequently, we have three labels (positive, negative and neutral) with the new distribution shown in Table 2, which contains 8,544 training samples and 2,210 testing samples, for a total of two labels.

Sample	Label	Quantity
Training	Positive	3610
Training	Negative	3310
Training	o	1624
Testing	Positive	1743
Testing	o	350
Testing	Negative	117

Table 2: New distribution of SST5 dataset

3.2.2 SemEval 2018

The purpose of this research is to evaluate the performance of language-independent representation in complex natural language processing tasks. As a result, we conducted experiments involving emotion classification. It is a method of demonstrating that language-independent representation is effective in a variety of tasks. In order to accomplish this, we train our model using the SemEval 2018 [27] dataset.

In this dataset, tweets from the English, Spanish, and Arabic languages are labeled and grouped together. It is further subdivided into individual tasks such as emotion intensity regression (El-reg), emotion intensity ordinal classification (El-oc), sentiment intensity (V-reg), sentiment analysis and ordinal classification (V-oc), and emotion classification (E-c). However, in our

thesis, we only address the E-c task. The main distinction between this task and the SST5 dataset is that it provides labels in the form of emotion and has multiple labels for each tweet. It is more complicated than having a single label for a sentence, as in SST5. Because each tweet may contain multiple emotions at the same time, and some emotions may contain multiple other emotions. Consider the emotion of surprise, which can be positive or negative depending on the context. Overall, the SemEval 2018 task 1EC includes six emotions: disgust, fear, sadness, surprise, joy, and anger, as shown in Table 3. However, in this study, we are only concentrating on four of them (anger, joy, disgust, surprise)

Text	disgust	fear	sadness	surprise	joy	anger
@Max_Kellerman it also helps that the majority of NFL coaching is inept. Some of Bill O'Brien's play calling was wow, ! #GOPATS	1	0	0	0	1	1
Accept the challenges so that you can literally even feel the exhilaration of victory.' – George S. Patton	0	0	0	0	1	0
it's pretty depressing when u hit pan on ur favourite highlighter	1	0	1	0	0	0
@BossUpJaee but your pussy was weak from what I heard so stfu up to me bitch . You got to threaten him that your pregnant .	1	0	0	0	0	1
Making that yearly transition from excited and hopeful college returner to sick and exhausted pessimist. #college	1	0	1	0	0	0

Table 3: Five randomly chosen sentences with their labels from SemEval 2018 task 1EC dataset

3.3 Machine translation

For the purpose of converting our datasets from the English language to other languages, we decided to use Google Translate. It is a service that provides translation of text from one language to another nearly instantly because they have sophisticated algorithms and billions of documents that have been expertly translated. However, this does not imply that translation is error-free or without flaws of any kind, which means that there is still room for improvement and that it is in the process of improving. For instance, historically, it used statistical machine translation until google introduced the Google Neural Translation System (GNMT) that uses state-of-the-art training methods to get tremendous improvements regarding the quality of machine translation [24]. The engineers at the company came up with a Recurrent Neural Network (RNN) that helped in learning the manner of mapping a sentence sequence as it is input in one language to the sequence when it is output in another language [76]. Google Translate now supports a total of 108 languages in total. Additionally, they create interfaces in the form of APIs, such as the Translation API[4], that make it easier to use for a variety of purposes, and we also used this interface to translate the data from one language to another.

[4]https://cloud.google.com/translate

3.4 Text preprocessing

Naturally, text preprocessing is a critical component of the field of natural language processing (NLP). It transforms text into a more desirable form, which allows it to be used in machine learning algorithms to produce better results. In our research, we are looking into the world of tweeters and movie reviews. Accordingly, we employ methods to clean and process text that is dependent on the context of the data. In this study, we will take a look at three steps that are commonly used in text preprocessing: 1) Text noise removal, 2) Text tokenization, and 3) Text normalization.

Primary cleaning operations are performed on the text to make it easier to read by a machine. This process involves removing unnecessary and distracting parts of the text to make it easier to read by a machine. Following that, the strings are divided into smaller tokens, from which large text can be divided into sentences, and sentences can be tokenized into small words. Finally, the text must be normalized, which means that all of the characters within the text must be lowercased.

3.4.1 Basic cleaning operations

We performed a series of steps to clean text, and the first is cleaning unnecessary things from the text:

- We are eliminating URLs from the text as they do not contain valuable information for the model.
- Eliminating symbol "#" from hashtags (e.g., #veryhappy).
- Eliminating username mentions as the name of a person is not providing valuable information regarding sentiment or emotion (e.g., @JoeBiden)
- We are eliminating emojis from the text. However, it is possible to convert emojis to corresponding emotional words. We just removed it in our experiment as it is not the main object of this study.

We also cleaned the symbols, such as ''', which appear after using a translator. Below, we gave a result of the translation dataset from English to the Dutch language.

> *The Rock is voorbestemd om de nieuwe " Conan '' van de 21e eeuw te worden en dat hij nog meer indruk zal maken dan Arnold Schwarzenegger, Jean-Claud Van Damme of Steven Segal.*

3.4.2 Text tokenization and normalization

Tokenization is the technique of breaking a text into smaller units referred to as tokens used in input processes like mining of text or parsing. It is used to comprehend patterns with readers that enable the achievement of tasks that include named entity recognition, POS tagging, and sentiment analysis [77], [78]. In this study, we used different tokenizers such as NLTK[5], and

[5]https://www.nltk.org/index.html

Moses[6].

NLTK is a subset of the Python programming language with utilities and functions that are prebuilt to make it easier to implement and use the program. When looking at the libraries with natural language computational and processing linguistics, it is one of the most used [79]). There are various techniques of using the NLTK tokenizer:

- Character tokenization - text is broken into characters
- Word tokenization - long text brokes into words
- Sentence tokenization - divides paragraphs into a list of sentences
- Whitespace tokenization - tokenizing whitespace through a string
- Word punctuation tokenization - uses a series of punctuations to tokenize sentences

We used NLTK with word tokenization method in all languages that we focus on this study. Then, we used MOSES tokenizer to implement one of our language-independent sentence representers.

The MOSES tokenizer comes in the toolkit called Moses, which is mainly used to separate words and punctuations from text, hence preserving some tokens that are regarded as exceptional, including dates and URLs [80]. As it does so, it normalizes the characters, hence making it easier to implement with any language.

Finally, we transformed all text to lowercase and removed all stopwords to normalize our dataset.

3.5 Representation learning

Following the completion of text preprocessing, we will proceed with feature extraction. In a baseline, we were employing fastText, which provides pre-trained embeddings for each language. In spite of the fact that it is considered to be a multilingual approach, we used monolingual pre-trained embedding, which was provided by FastText, and trained the machine learning model for each language separately. Therefore, this approach, to some extent, inefficient in terms of time and memory planning in the future. As an alternative, we also used sentence embeddings such as the Language Agnostic Sentence Representation (LASER)[17] and the Language-agnostic BERT sentence embedding (LaBSE)[18].

3.5.1 Langauge Agnostic Sentence Representation

The vector representations of sentences employed by LASER are universal in that they can be used regardless of the language of the input or the NLP task. The tool translates a sentence in any language into a location in a high-dimensional space, with the goal of putting the identical statement in any language in the same neighborhood as the converted sentence. It is possible because of the technology of neural machine translation, known as the encoder-decoder method that we explained in Chapter 2.

[6]https://github.com/luismsgomes/mosestokenizer

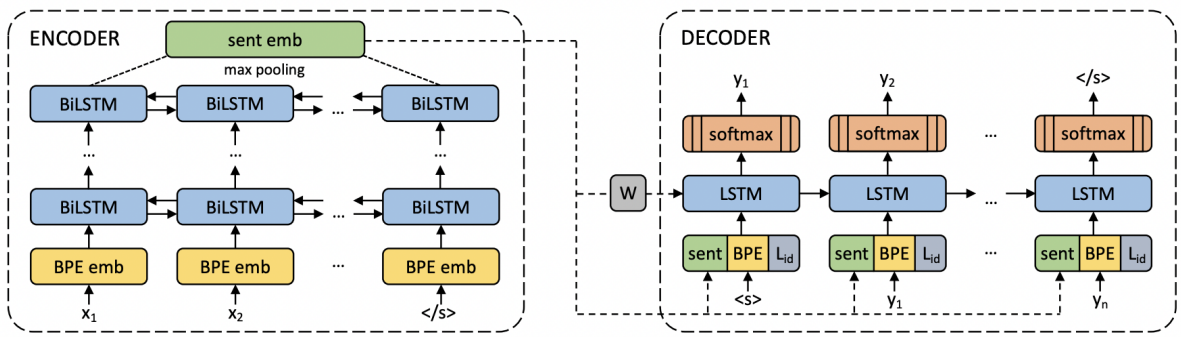

Figure 8: LASER architecture [17]

Based on Schwenk's[25] work, the LASER's system architecture is depicted in Figure 8. A five-layer bidirectional LSTM (long short-term memory) network serves as the encoder. In contrast to neural machine translation, the LASER does not use an attention mechanism and instead represents the input sentence with a 1,024-dimension fixed-size vector. Sentence embeddings are constructed, as shown in the example, by applying a max-pooling operation to a BiLSTM encoder's output to obtain a sentence embedding. These embeddings are used to initialize the decoder LSTM via a linear transformation and are concatenated to its input embeddings at each step. We can also notice that there is no other connection between the encoder and the decoder because the sentence embedding has to capture all relevant information from the input sequence. The encoder does not receive a clear signal indicating what language the input is in; it is encouraged to learn representations independent of the input language. A language ID embedding, on the other hand, specifies the language to generate and is concatenated with the input and sentence embeddings at each time step in the decoding process. LASER utilizes a joint byte-pair encoding (BPE) vocabulary with 50k operations, which learns sub-words by observing frequent words in n-grams [17]. It is required because the system only has one encoder and decoder where all languages are involved.

The LASER encoder takes advantage of the parallel corpora data, which means that they benefit from the associations that exist between parallel sentences. The goal is to provide the model with opportunities to translate from the source language to any other target languages based on the knowledge gained. It is possible because the model can predict masked tokens in both the source and target languages at the same time. For example, consider the sentences below:

English: *Dramas like this* ***[MASK]*** *it human.*

Russian: *Подобные драмы* ***[MASK]*** *его человечным.*

It masked the word "make" in the source language, and "делают" in the target language. The process of masking tokens happens randomly and for all words in sentences. Furthermore, the model can leverage information from both language context to predict masked words. Consequently, the model able to calculate how similar the sentence and can extend it to measure the relationship between different language datasets. Thus, it is a handy function that we also harnessed in our study and presented in a coming section.

In total, LASER was trained using a dataset of 93 different languages. These languages were written in 28 different scripts and belonged to 30 different families of languages. Through the use of a single encoder, BiLSTM, they were able to train the vocabulary of all languages using publicly available datasets. Despite the fact that the original paper also provides a classifier[17], this study only makes use of the LASER pre-trained model as a feature extractor in the future.

3.5.2 Language Agnostic BERT Sentence Embedding

LaBSE is a multilingual BERT embedding model which supports 109 languages ([18]). Six billion bilingual sentence pairs and 17 billion monolingual sentences were used to train it.

The dual encoder architecture encodes the source and target text separately using a shared transformer embedding network, as illustrated in figure 9. The model uses bidirectional dual encoders with additive margin softmax loss with in-batch negative sampling [81], as shown in following equation:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\frac{e^{\phi(x_i,y_i)^{-m}}}{e^{\phi(x_i,y_i)^{-m}} + \sum_{n=1,n\neq 1}^{N} e^{\phi(x_i,y_n)}}$$

$\phi(x, y)$ given as space similarity of x and y embeddings. Even when $\phi(x_i, y_i)$ is discounted by margin m, the loss attempts to rank y_i, the true translation of x_i, overall $N - 1$ alternatives in the same batch. The final loss function adds the source to target $\mathcal{L}$ and target to source $\mathcal{L}^{'}$ loss functions, as shown below:

$$\overline{\mathcal{L}} = \mathcal{L} + \mathcal{L}^{'}$$

Large training batch sizes benefit cross-lingual embedding models trained within batch negative samples [82]. Then, a transformer encoder is used [15]. On the monolingual data and bilingual translation pairs, the encoder is pre-trained with Masked Language Model (MLM) [16] and Translation Language Model (TLM) [83] training. Finally, the model trained with a L layer transformer encoder using a three-stage progressive stacking algorithm [84]. As with the LASER model, the LaBSE pre-trained model is being used as a feature extractor.

3.6 Text classifier

Different machine learning algorithms can be employed when it comes to sentiment analysis and emotion classification. Similarly, Support Vector Machines (SVMs) offers a simple and powerful approach to solve practical problems. They are acknowledged extensively by many machine learning experts globally due to their propensity to produce significantly accurate data using minimal computation power[85]. This thesis used SVM with Gaussian kernel as a machine-learning algorithm to conduct experiments due to simplicity and memory efficiency.

3.6.1 Support Vector Machines (SVM)

The SVM machine language algorithm makes use of a specific hyperplane or line in an N-dimensional space that vividly classifies the data points, where N represents the total sum of the features. When separating two groups or classes of data points, a number of different hyperplanes or lines can be used to accomplish this. The Support Vector Machine learning

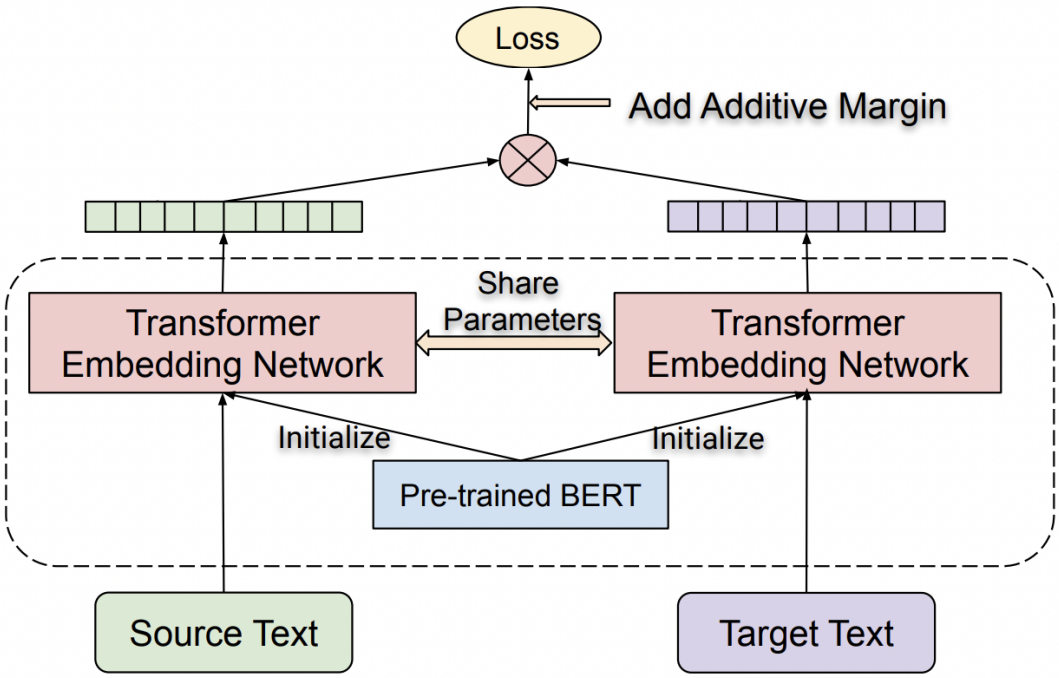

Figure 9: LaBSE architecture [18]

algorithm necessitates the discovery of the point that is representative of both classes and is closest to the line. These specific points are referred to as Support vectors[86]. Subsequently, it is imperative to calculate the distance between the support vectors and the hyperplane. This distance is referred to as the margin. As a result, the primary objective is to identify a line that has the optimal margin. The optimal margin is defined as the distance between the data points from both groups that is the greatest possible. So optimizing the margin distance provides some level of reinforcement to ensure that data points can be classified more confidently in the feature, thereby increasing their likelihood of being classified correctly. The term "optimal hyperplane" refers to an instance in which the line for which the margin is at its maximum is the best line possible.

3.6.1.1 Parameter tuning

The SVM has several parameters to control training strategies. For instance, parameters such as gamma and *c* define the decision boundaries of the classifier. However, we have not focused on fine-tuning these parameters. Instead, we used CalibratedClassifierCV[7] in order to tackle the problem regarding calibration of probabilities of class. Not all machine algorithms can handle it, and SVM is one of them where a class is uncalibrated because it cannot predict class probabilities natively. CalibratedClassifierCV estimates the optimal parameters for the classifier and calibrates it. In our experiment, we used the cross-validation method of CalibratedClassifierCV with 5-folds. Typically, the application of SVM is restricted to two-dimensional cases in which the classifier can distinguish between labels using a straight line. However, it is incapable of dealing with problems with a large number of dimensions. As a result, it makes use of Kernel

[7]https://scikit-learn.org/stable/modules/generated/sklearn.calibration.CalibratedClassifierCV.html

Trick, and there are many of them.

3.6.1.2 Kernel

A *kernel* is a mathematical function that transforms input feature vectors into the required dimensional space to process them more effectively—there various types of the kernel.

- Basic formulation of kernel function:

$$K(X, y) =< f(x), f(y) >$$

- The linear function uses the dot product of features and makes a prediction with the classifier's decision boundary in another dimensional space.

$$K(x, y) = x_i \cdot x_j$$

- Radial Basic Function (RBF) is another standard kernel used in SVM. RBF kernel is a function that uses the Gaussian function to separate classes.

$$K(x, y) = exp(-\gamma |x_i - x_j|^2)$$

In the thesis, we used RBF as it is most generally used because of high performance.

3.6.1.3 Multiclass classification

Multiclass classification is predicting classes that have more than two labels. The SVM does not support multiclass classification as it is binary-oriented and can classify a maximum of two classes. However, SVM handles it by breaking down the multiclass problem into many binary classifications. There two techniques to achieve that:

1. One-to-one classification is creating a binary classifier for every two classes separately. It is computationally expensive as it will have N(N-1)N classifiers.

2. One-to-all classification is creating a binary classifier by training one class against others. It means that other classes will be assigned as one same label.

We used LASER and LaBSE as a feature extractor, trained RBF kernel SVM classifier. The Gaussian itself shows that it can handle dense embeddings, and ZyLAB's experience also proves it. Furthermore, we used one-to-all classification in our experiment as we find it most suitable from perspective time and memory.

3.7 Evaluation

F1-Score was used as the evaluation metric as our tasks considered as a classification problem. It is a form of measurement used to gauge the overall efficiency levels of a particular classifier in terms of accomplishing its set goals and objectives. Primarily, it combines both recall and precision for every tag involved. The F1 score differs from other kinds of measuring tools, such as accuracy, in that it produces more efficient results by vindicating any disparities during the dissemination of text amongst tags. Therefore, the F1 Score function becomes quite essential when seeking a balance between recall and precision.

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

A recall is often used to determine the percentage of texts predicted by a particular classifier from a more significant sum of texts that it should have predicted for those particular tags. Thus, it calculates the number of Actual Positives that the specific model captures by identifying it as a True positive (TP) [87]. For example, suppose a bank predicts a counterfeit transaction as a genuine one. It can encounter various challenges, assuming the counterfeit transaction is the True Positive, and the genuine one is the Predicted Negative. Incorporating a synonymous analogy, it is pretty clear that the recall refers to the metric model applied to identify the best model in instances where a False-negative (FN) is associated with high cost.

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

Unlike Recall, Precision can be described as the actual percentage of texts that the classifier model identified correctly from the general sum of texts predicted from a particular tag. Moreover, this concept relates to determining how accurate or precise the model is from the various predicted positives. It is also an effective means of measuring and determining high costs regarding False Positives (FP). For instance, a non-spam email sent and detected as a spam email can be termed a False Positive. In such a scenario, the email owner will most likely lose several essential emails if the Precision is relatively low for the particular spam-detection model.

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

3.7.1 Micro Averaged F1 Score

In this thesis, we are using a micro average F1 score which aggregates all labels contributions to calculate the average metric. But, it can completely dominate the poor performance of smaller classes if the majority class performs significantly well. Of course, this is unfair to minority groups because their low individual scores would be masked by a high overall metric. Nevertheless, we decide to take the micro average F1 score, as it is a preferable way to handle a class imbalance in a multi-class classification setup. Same as we have classification with one-to-all context with imbalanced classes.

$$F1_{\mu} = \frac{\sum_{i=0}^{N-1} F1_i}{|all|}$$

Where the N is a number of classes, and *all* is the number of instances.

4 Results

4.1 Baseline

In this thesis, we study several language-independent representations and conduct experiments to understand better how they perform in various natural language processing tasks. However, before we start, we wanted to establish our benchmark for each NLP task and use that as a starting point for our investigation. From our perspective, the most appropriate way, to begin with, was the fastText word embeddings. It provides a single pre-trained model for each language, and there is no use of shared knowledge between the languages in this system. Furthermore, it is language sensitive and can only be used in the dataset with the same language as the pre-trained model. As a result, it seemed the most straightforward and logical strategy because we train a classifier with only one language and evaluate it with the corresponding language.

To use the fastText pre-trained model, we need a separate dataset for each of the 12 languages listed above. Both SST5 and SemEval 2018 task 1EC datasets provide datasets in several languages. However, we selected only English as the source language in order to bypass complications in our methodology. It is much easier to follow the data transfer when there is only one source. We were able to complete this knowledge transfer task using Google Translation, and it is carried out by using the cloud translation API. First, we designated English as the source language and the other 12 languages as the target languages. However, before we start translating, we divided the dataset into buckets, each containing 100 sentences as a separate item in a list. It is done to avoid having too many requests in google service. One thing to keep in mind is that when we send a list of sentences to the cloud, each item in the list is considered and translated separately. It means that the context of the bucket does not affect the context of sentences. Finally, we able to translate our datasets to all languages we considered as target languages. The result of translation for Russian and Spanish languages are shown below:

> **English** - source language: *Australian actor/director John Polson and award-winning English cinematographer Giles Nuttgens make a terrific effort at disguising the obvious with energy and innovation.*
>
> **Russian** - target language: *Австралийский актер и режиссер Джон Полсон и отмеченный наградами английский кинематографист Джайлс Наттгенс прилагают огромные усилия, чтобы замаскировать очевидное с помощью энергии и новаторства.*
>
> **Spanish** - target language: *El actor y director australiano John Polson y el galardonado director de fotografía inglés Giles Nuttgens hacen un tremendo esfuerzo por disfrazar lo obvio con energía e innovación.*

As we can see from the example, Google Translate was able to provide accurate translations for both target languages used. It can even recognize symbols such as the "/" symbol, which in English represents the word "or," and replace it with the appropriate word in the target language. However, in terms of word order, the translator maintained the same structure as in

English, which, for example, may not sound natural in Russian but still conveys the primary meaning. Finally, we used the machine-translated datasets to train our SVM classifier, which resulted in the creation of a model for each language. The experimental results of those classifiers will be presented in the following section.

Language	Label	Precision	Recall	F1
English	Positive	**0.74**	**0.75**	**0.76**
English	Negative	**0.76**	**0.57**	**0.65**
Dutch	Positive	**0.74**	0.68	0.71
Dutch	Negative	0.74	0.54	0.63
German	Positive	0.73	0.67	0.70
German	Negative	0.72	0.52	0.60
Swedish	Positive	0.72	0.68	0.70
Swedish	Negative	0.71	0.50	0.59
Russia	Positive	0.72	0.69	0.70
Russia	Negative	0.71	0.52	0.60
Slovak	Positive	0.73	0.65	0.69
Slovak	Negative	0.71	0.51	0.59
Polish	Positive	**0.74**	0.66	0.70
Polish	Negative	0.74	0.53	0.62
Czech	Positive	**0.74**	0.66	0.70
Czech	Negative	0.71	0.52	0.60
Italian	Positive	0.73	0.69	0.71
Italian	Negative	0.73	0.52	0.61
French	Positive	**0.74**	0.66	0.70
French	Negative	0.70	0.52	0.59
Portuguese	Positive	0.73	0.67	0.70
Portuguese	Negative	0.72	0.55	0.62
Spanish	Positive	0.73	0.69	0.71
Spanish	Negative	0.72	0.54	0.62

Table 4: Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using FastText word embeddings.

4.1.1 FastText results

We used a gaussian kernel to train the SVM classifier and only ran the test once to evaluate the results. Then, we obtained micro averaged F1 score, recall, and precision results and recorded them in tables. In the following sections, we refer to F1 score rather than the micro average F1 score. The meaning, however, remains the same. To be clear, the results of the neutral label are not presented in this thesis because we are primarily interested in the labels on which we are concentrating our efforts.

Language	Label	Precision	Recall	F1
English	Anger	**0.73**	**0.60**	**0.66**
English	Joy	**0.84**	**0.68**	**0.75**
English	Disgust	**0.70**	**0.57**	**0.63**
English	Surprise	0.50	0.01	0.01
Dutch	Anger	0.71	0.55	0.62
Dutch	Joy	0.81	0.63	0.71
Dutch	Disgust	0.67	0.52	0.58
Dutch	Surprise	0.33	0.01	0.01
German	Anger	0.70	0.58	0.62
German	Joy	0.82	0.64	0.72
German	Disgust	0.67	0.52	0.58
German	Surprise	0.00	0.00	0.00
Swedish	Anger	0.69	0.54	0.61
Swedish	Joy	0.81	0.62	0.71
Swedish	Disgust	0.65	0.52	0.58
Swedish	Surprise	0.00	0.00	0.00
Russia	Anger	0.70	0.58	0.63
Russia	Joy	0.81	0.62	0.70
Russia	Disgust	0.67	0.54	0.60
Russia	Surprise	0.00	0.00	0.00
Slovak	Anger	0.65	0.48	0.56
Slovak	Joy	0.79	0.60	0.68
Slovak	Disgust	0.63	0.47	0.53
Slovak	Surprise	0.00	0.00	0.00
Polish	Anger	0.68	0.52	0.59
Polish	Joy	0.80	0.61	0.69
Polish	Disgust	0.64	0.49	0.55
Polish	Surprise	0.00	0.00	0.00
Czech	Anger	0.68	0.52	0.59
Czech	Joy	0.80	0.62	0.70
Czech	Disgust	0.63	0.47	0.53
Czech	Surprise	0.00	0.00	0.00
Italian	Anger	0.72	0.56	0.63
Italian	Joy	0.82	0.62	0.71
Italian	Disgust	0.68	0.53	0.59
Italian	Surprise	0.00	0.00	0.00
French	Anger	0.71	0.56	0.63
French	Joy	**0.84**	0.62	0.71
French	Disgust	0.67	0.52	0.59
French	Surprise	0.00	0.00	0.00
Portuguese	Anger	0.72	0.56	0.63
Portuguese	Joy	0.83	0.64	0.72
Portuguese	Disgust	0.66	0.52	0.58
Portuguese	Surprise	0.00	0.00	0.00
Spanish	Anger	**0.73**	0.58	0.65
Spanish	Joy	**0.84**	0.64	0.73
Spanish	Disgust	0.68	0.54	0.60
Spanish	Surprise	0.00	0.00	0.00

Table 5: Micro averaged F1 scores on the SemEval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained SemEval 2018 task 1EC training dataset by using FastText word embedding.

Table 4 shows the results of the SST5 dataset for each of the 12 languages in which we used it. In this case, we can see that the F1-score for the positive label is on average 0.71, with a maximum score of 0.76 in English and a minimum score of 0.69 in Slovak, on average. The average F1-score for a negative label is 0.61, with a maximum score of 0.65 in English, and the maximum score for a negative label is 0.61 in Spanish. The bare minimum of 0.59 points was shared by three languages: Swedish, Slovak, and French, respectively. The results of the classifier are nearly identical in all languages, giving the impression that Google translation is effective once more. After that, let us take a closer look at precision and recall. We can see that recall is generally lower than precision, with the exception of English with a positive label, which maintains the same level of precision and recall at 0.74 and 0.75, respectively. The precision for positive numbers is 0.73 on average, and for negative numbers, it is 0.72.

On the other hand, recall is 0.67 for positive labels and 0.52 for negative labels, which is lower than precision for 0.06 points and 0.2 points, respectively. We can also see that the negative label has a lower overall score than the positive label when compared to the positive label. It is possible that this is due to the fact that fastText word embeddings are unable to handle things such as negations. Typically, it is dealt with during the text pre-processing stage by replacing negations with words that have similar meanings. For example, the word **"can't"** could be substituted with the word **"cannot,"** which is the most basic example.

The result of SemEval 2018 task 1EC in Table 5 clearly shows that the gap between precision and recall remains unchanged as it was in SST5. The "angry" label has an average precision of 0.7 and recall of 0.55. Next, "disgust" has a precision of 0.66 and a recall of 0.51. Then, 0.82 precision and 0.62 recall have the label "joy". Only label "surprise" has a recall and precision of zero, as previously stated. It could be due to the complexities of this emotion.

langs	de	nl	sv	en	es	fr	it	pt	pl	ru	cs	sk	avg
de	0.00	1.60	1.58	3.56	1.79	8.74	4.08	1.88	1.85	2.67	1.58	1.93	2.84
nl	2.04	0.00	2.28	4.11	2.36	8.33	4.23	2.34	2.73	3.57	3.44	3.32	3.52
sv	1.80	1.91	0.00	4.17	1.73	7.84	3.76	1.94	2.11	2.69	2.93	3.65	3.14
en	5.30	5.45	6.19	0.00	3.66	6.25	4.48	4.35	6.29	7.14	6.07	5.83	5.55
es	1.98	2.01	1.74	2.40	0.00	5.57	2.35	0.94	1.80	2.13	2.31	2.94	2.38
fr	21.43	23.36	24.67	21.59	18.56	0.00	15.43	19.94	25.76	26.36	24.58	20.45	22.01
it	7.61	8.52	8.70	7.54	5.47	6.23	0.00	6.11	8.95	9.57	8.85	7.85	7.76
pt	2.04	2.08	1.88	2.76	1.03	6.29	2.52	0.00	1.99	2.68	2.39	2.79	2.59
pl	2.45	2.50	2.15	4.72	2.18	10.14	4.76	2.38	0.00	2.22	2.29	2.55	3.49
ru	2.75	2.97	2.73	5.31	2.33	11.12	5.27	2.54	1.81	0.00	2.68	3.24	3.89
cs	1.64	2.48	1.85	3.86	1.94	9.38	4.25	1.94	1.53	2.12	0.00	1.21	2.93
sk	1.72	2.27	2.49	4.27	2.12	9.40	4.53	1.97	1.59	2.15	0.98	0.00	3.05
avg	4.61	5.02	5.12	5.84	3.93	8.12	5.06	4.21	5.13	5.76	5.28	5.07	5.26

Table 6: SST5 all language pairs similarity errors in percentage (langs - languages, de-German, nl-Dutch, sv-Swedish, en-Englis, es-Spanish, fr-French, it-Italian, pt-Portuguese, pl-Polish, ru-Russia, cs-Czech, sk-Slovak).

Nonetheless, we can clearly see the difference in results between English and other languages in this task. For example, the F1-score for the angry label is 0.66 in English, while the other

languages average 0.61, with the exception of Spanish, which has 0.65. The F1 score has dropped by at least 0.03 points. When we consider the context in which the dataset was translated, we can conclude that it is not a significant drop.

4.2 LASER

We already know that the LASER model keeps the sentences in common space for all 93 languages it supports. For our experimentation, we are only using 12 languages, and we have translated datasets for each language. Consequently, we need to encode them with a pre-trained LASER model. The encoding process is covered following steps: 1) We used the fastBPE[8] package that uses neural machine translation to make subword units from rare words. 2) We used LASER pre-trained model implemented by fairseq[9] to encode the datasets, which at the end returned in $N \times 1024$ vector space in NumPy[10] array format. The N is the number of sentences in the dataset. After completion of encoding, we had 12 languages encoded datasets, which contains training and testing samples. This section examines the similarity of our translated dataset and gives the results of the SVM classifier with LASER embedding.

langs	de	nl	sv	en	es	fr	it	pt	pl	ru	cs	sk	avg
de	0.00	2.49	3.88	4.33	3.55	5.34	2.92	2.54	3.45	3.60	4.05	4.24	3.67
nl	2.87	0.00	3.45	4.14	2.97	4.75	2.81	2.09	3.60	3.71	4.08	4.40	3.53
sv	3.83	3.00	0.00	4.75	3.95	5.44	3.50	3.13	3.48	3.76	3.60	4.04	3.86
en	6.04	4.87	6.42	0.00	4.28	5.09	3.89	3.93	6.03	6.42	6.42	6.57	5.45
es	3.35	2.54	3.99	3.39	0.00	4.28	2.47	2.09	3.66	3.88	4.62	4.49	3.52
fr	15.94	15.03	17.36	13.47	13.37	0.00	10.87	12.75	16.22	16.89	16.41	16.32	14.97
it	4.91	3.99	5.47	4.49	3.58	3.92	0.00	2.90	4.91	5.15	5.48	5.47	4.57
pt	2.60	1.93	3.44	3.10	2.30	3.66	1.73	0.00	3.00	3.69	3.89	4.12	3.04
pl	3.36	2.88	3.71	4.53	3.42	5.12	2.98	2.43	0.00	3.04	3.14	3.38	3.46
ru	3.61	3.04	3.86	4.71	3.57	5.75	3.14	2.54	3.07	0.00	3.29	3.48	3.64
cs	3.99	3.60	3.47	4.59	4.07	5.62	3.39	3.06	3.04	3.31	0.00	2.22	3.67
sk	3.67	3.29	3.35	4.46	3.99	5.57	3.19	2.97	3.04	3.26	1.89	0.00	3.52
avg	4.93	4.24	5.31	5.09	4.46	4.96	3.72	3.68	4.86	5.15	5.17	5.34	4.74

Table 7: SemEval 2018 task 1EC all language pairs similarity errors in percentage (langs - languages, de-German, nl-Dutch, sv-Swedish, en-Englis, es-Spanish, fr-French, it-Italian, pt-Portuguese, pl-Polish, ru-Russia, cs-Czech, sk-Slovak).

4.2.1 Similarity analysis

Before we get into the interpretation of the results, we would like to share the results of the similarity error analyses we performed to see how well the translation went. We accomplished this with the help of LASER and the library faiss[11], which are both efficient in their ability to find similarities and provide errors between languages in the form of percentages. The similarity error is the vectors that are compared using Euclidean distances or dot products. Then, it is

[8]https://github.com/glample/fastBPE
[9]https://github.com/pytorch/fairseq
[10]https://numpy.org/
[11]https://github.com/facebookresearch/faiss

the least distance between two query vectors or the highest dot product between two query vectors similar to each other in the form of a percentage. Tables 6 and 7 show the percentage of errors made by each language pair when compared to other languages. Based on two of these tables, we can conclude that the vast majority of errors are less than 9 percent; only in the French language does it reach 18 percent in SemEval 2018 task 1EC and SST5 does it reach 27 percent. It is possible that this is due to the complexity of the French language, which might have unique language properties that Google Translator cannot handle. We translated text from English to French and then the exact same text from French to English in order to figure out what might be causing the significant error to occur. As an illustration, consider the following three sentences that were chosen at random:

Example #1:

> **English** - source: *If you love reading and/or poetry , then by all means check it out .*
>
> **French** - target/source: *Si vous aimez la lecture et / ou la poésie, alors jetez-y un œil.*
>
> **English** - target: *If you like reading and / or poetry, then check it out.*

Example #2:

> **English** - source: *The path Ice Age follows most closely , though , is the one established by Warner Bros. giant Chuck Jones , who died a matter of weeks before the movie 's release.*
>
> **French** - target/source: *Le chemin que Ice Age suit de plus près, cependant, est celui établi par le géant de Warner Bros. Chuck Jones, décédé quelques semaines avant la sortie du film.*
>
> **English** - target: *The path that Ice Age is most closely following, however, is one established by the Warner Bros. giant. Chuck Jones, who died a few weeks before the film's release.*

Example #3:

> **English** - source: *For those who pride themselves on sophisticated , discerning taste , this might not seem like the proper cup of tea , however it is almost guaranteed that even the stuffiest cinema goers will laugh their *** off for an hour-and-a-half.*
>
> **French** - target/source: *Pour ceux qui sont fiers de leur goût sophistiqué et perspicace, cela peut ne pas sembler être la bonne tasse de thé, mais il est presque garanti que même les cinéphiles les plus étouffants riront de leur *** pendant une heure et demie.*
>
> **English** - target: *For those who pride themselves on their sophisticated and insightful taste, this may not sound like the right cup of tea, but it's almost guaranteed that even the most stuffy moviegoers will laugh at their *** for an hour and a half.*

In general, it appears that Google Translation is adequate for most purposes, with only a few minor errors. Then, it makes it more challenging to determine why the similarity analysis yields such a high error rate when compared to other languages.

Language	Label	Precision	Recall	F1
English	Positive	0.76	0.70	0,73
English	Negative	0.74	0.60	0,66
Dutch	Positive	0.78	0.68	0,73
Dutch	Negative	0.71	0.68	0,69
German	Positive	0.76	0.71	**0,74**
German	Negative	0.72	**0.70**	0,71
Swedish	Positive	0.77	0.69	0,73
Swedish	Negative	0.71	0.67	0,69
Russia	Positive	0.78	0.71	**0,74**
Russia	Negative	0.71	**0.70**	0,71
Slovak	Positive	0.76	0.71	**0,74**
Slovak	Negative	0.72	0.67	0,70
Polish	Positive	0.78	0.71	**0,74**
Polish	Negative	0.73	**0.70**	**0,72**
Czech	Positive	0.74	**0.72**	0,73
Czech	Negative	0.76	0.60	0,67
Italian	Positive	0.77	0.68	0,72
Italian	Negative	0.73	0.63	0,68
French	Positive	**0.80**	0.63	0,70
French	Negative	**0.77**	0.53	0,62
Portuguese	Positive	0.77	0.70	0,73
Portuguese	Negative	0.72	0.67	0,69
Spanish	Positive	0.77	0.71	**0,74**
Spanish	Negative	0.72	0.67	0,62

Table 8: Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using LASER embeddings.

4.2.2 LASER results

Finally, the SVM classifier with a gaussian kernel that was trained only with English encoded embeddings and tested in other encoded language datasets, including English itself, was found to be effective. Looking at the results of LASER for SST5 in Table 8, we can see that the results have significantly improved. For example, the F1-score improved outcomes by 7 percent when compared to the baseline, while the precision and recall improved by 2.97 percent and 11,59 percent, respectively. We obtained these results by using only a training model with English encoded embedding as the only feature. As of now, the average F1-score is 0.73 in the positive category and 0.68 in the negative category. The results of task 1EC of SemEval 2018 are then shown in Table 9. Similarly, we can see an improvement in this task; for example, the F1-score has improved by 7.94 percent, recalls have improved by 9.95 percent, and precision has improved by 3.07 percent. However, the emotion "surprise" remains at zero in all languages, which is the same as the results of FastText embedding. Therefore, we removed it from the table because there was no helpful information contained within. Overall, we can conclude that LASER embedding performs significantly better than our baseline.

Language	Label	Precision	Recall	F1
English	Anger	0.74	0.60	0.66
English	Joy	0.84	**0.74**	**0.79**
English	Disgust	0.70	0.57	0.63
Dutch	Anger	0.75	0.58	0.66
Dutch	Joy	0.85	0.72	0.78
Dutch	Disgust	0.71	0.58	**0.64**
German	Anger	0.76	0.57	0.65
German	Joy	0.86	0.70	0.77
German	Disgust	0.70	0.55	0.62
Swedish	Anger	0.75	0.57	0.65
Swedish	Joy	0.84	0.72	0.77
Swedish	Disgust	0.70	0.56	0.62
Russia	Anger	0.75	**0.61**	**0.67**
Russia	Joy	**0.86**	0.70	0.77
Russia	Disgust	0.69	0.58	0.63
Slovak	Anger	0.75	0.59	0.66
Slovak	Joy	0.85	0.72	0.78
Slovak	Disgust	0.71	0.56	0.62
Polish	Anger	**0.77**	0.58	0.66
Polish	Joy	0.85	0.70	0.77
Polish	Disgust	0.70	0.56	0.62
Czech	Anger	0.75	0.57	0.65
Czech	Joy	0.85	0.71	0.77
Czech	Disgust	**0.72**	0.55	0.62
Italian	Anger	0.74	0.60	0.66
Italian	Joy	0.83	0.72	0.77
Italian	Disgust	0.68	**0.60**	**0.64**
French	Anger	0.76	0.55	0.64
French	Joy	0.83	**0.74**	0.78
French	Disgust	0.70	0.55	0.62
Portuguese	Anger	0.75	0.60	**0.67**
Portuguese	Joy	0.84	0.71	0.77
Portuguese	Disgust	0.69	0.58	0.63
Spanish	Anger	0.73	0.59	0.65
Spanish	Joy	**0.86**	0.71	0.78
Spanish	Disgust	0.68	0.56	0.62

Table 9: Micro averaged F1 scores on the Semeval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained Semeval 2018 task 1EC training dataset by using LASER embeddings.

Type	Metrics	Germanic	Romance	Slavic	Mix
Baseline	F1-score	**10.95**	9.94	10.19	10.82
Baseline	Precision	2.98	**3.26**	3.15	3.04
Baseline	Recall	**18.50**	16.15	16.98	17.87
LASER	F1-score	**3.54**	2.60	2.84	3.43
LASER	Precision	0.00	0.28	0.17	0.06
LASER	Recall	**6.18**	4.08	4.82	5.63

Table 10: The results of LASER-FG compare to the SST5 baseline and LASER in percentage.

4.3 LASER Family Group (LASER-FG)

Based on the performance of the LASER, we came up with the idea of training a classifier with multiple encoded datasets to try to improve its accuracy. We believe that it will improve the performance of the results classifier because some studies have shown that classifier performance is significantly better across similar languages [88]. In addition, it is attractive to determining whether or not language properties have an impact on the model's results. That is one of the reasons to try to train the model using language family groups as a basis for training.

Type	Metrics	Germanic	Romance	Slavic	Mix
Baseline	F1-score	19.29	19.60	**20.78**	19.42
Baseline	Precision	7.37	8.26	8.63	**12.00**
Baseline	Recall	28.73	28.04	**30.00**	26.81
LASER	F1-score	10.51	10.80	**11.89**	10.63
LASER	Precision	4.17	5.03	5.39	**8.66**
LASER	Recall	17.08	16.45	**18.23**	15.34

Table 11: The results of LASER-FG compare to the SemEval 2018 task 1EC baseline and LASER in percentage.

In total, we had four groups, including Germanic, Romance, Slavic, and Mixed. We decided to collect different language families such as English, Russian and Spanish into the mixed group. We believe that it has the potential to improve the model's performance because it could cover a broader range of language parameters. Table 10 and Table 11 show how the family group model performed compared to baseline and SVM model, which trained with only English LASER encoded dataset; we name as LASER. We can see that, on average, the F1-score, precision, and recall in SST5 among the different family groups similar but with a slight deviation. For instance, the Germanic group has made significant gains in both F1-score and recall, with increases of 10.95 percent and 18.5 percent, respectively, compared to the baseline. The least F1 score gained Romance group with 9.94 percent increase. However, it has the most increase in precision. On the other hand, the LASER-FG show growth in F1-score compared to LASER, with the most significant rise occurring in the Germanic group once again. It has a 3.54 percent increase. In general, we can understand that this improvement comes from an increase in recall in all family groups on average for 5.1 percent. However, precision has no significant change.

Language	Label	Germanic			Romance			Slavic			Mix		
		Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
English	Positive	0.75	0.74	0.75	0.81	0.60	0.69	0.8	0.65	0.72	0.74	0.74	0.74
English	Negative	0.75	0.62	0.68	0.72	0.68	0.70	0.74	0.65	0.69	0.76	0.62	0.68
Dutch	Positive	0.76	0.75	0.76	0.74	0.74	0.74	0.74	0.75	0.75	0.75	0.75	0.75
Dutch	Negative	0.73	0.68	0.70	0.74	0.66	0.7	0.73	0.67	0.70	0.74	0.69	0.72
German	Positive	0.75	0.77	0.76	0.74	0.76	0.75	0.74	0.77	0.76	0.75	0.76	0.75
German	Negative	0.73	0.69	0.71	0.76	0.67	0.71	0.74	0.67	0.7	0.74	0.68	0.71
Swedish	Positive	0.77	0.76	0.76	0.75	0.76	0.76	0.75	0.78	0.76	0.76	0.75	0.76
Swedish	Negative	0.73	0.68	0.70	0.73	0.65	0.68	0.73	0.66	0.69	0.73	0.68	0.70
Russia	Positive	0.77	0.73	0.75	0.76	0.74	0.75	0.76	0.77	0.76	0.77	0.75	0.76
Russia	Negative	0.72	0.69	0.71	0.74	0.68	0.71	0.73	0.68	0.70	0.73	0.70	0.72
Slovak	Positive	0.76	0.75	0.76	0.76	0.75	0.75	0.78	0.76	0.77	0.77	0.76	0.77
Slovak	Negative	0.73	0.68	0.70	0.74	0.66	0.70	0.72	0.67	0.69	0.73	0.67	0.70
Polish	Positive	0.77	0.77	0.77	0.75	0.77	0.76	0.77	0.79	0.78	0.77	0.78	0.77
Polish	Negative	0.75	0.70	0.72	0.75	0.67	0.71	0.74	0.69	0.71	0.75	0.71	0.73
Czech	Positive	0.75	0.76	0.75	0.82	0.63	0.71	0.78	0.74	0.76	0.75	0.76	0.75
Czech	Negative	0.76	0.74	**0.75**	0.73	0.68	0.70	0.75	0.64	0.69	0.76	0.58	0.66
Italian	Positive	0.76	0.75	0.75	0.75	0.76	0.76	0.77	0.74	0.75	0.77	0.74	0.75
Italian	Negative	0.73	0.66	0.69	0.75	0.68	0.71	0.72	0.67	0.69	0.72	0.68	0.70
French	Positive	0.76	0.72	0.74	0.75	0.75	0.75	0.77	0.71	0.74	0.77	0.71	0.74
French	Negative	0.74	0.64	0.69	0.75	0.66	0.70	0.73	0.65	0.68	0.72	0.67	0.69
Portuguese	Positive	0.75	0.76	0.75	0.76	0.76	0.76	0.77	0.75	0.76	0.75	0.76	0.76
Portuguese	Negative	0.75	0.68	0.71	0.74	0.68	0.71	0.73	0.66	0.69	0.75	0.68	0.71
Spanish	Positive	0.77	0.78	0.77	0.75	0.77	0.76	0.78	0.76	0.77	0.77	0.78	**0.78**
Spanish	Negative	0.74	0.67	0.70	0.74	0.67	0.70	0.74	0.67	0.70	0.74	0.68	0.71

Table 12: Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 family groups training dataset using LASER embedding.

By taking a closer look at the results of LASER-FG, shown in Table 12, we can recognize that groups such as Romance poorly work compare to other groups in the English and Czech languages. For instance, the positive label has a 0.69 and 0.71 F1 score—generally, the classifier results higher than 0.74 F1 scores in the positive label. We can also see that the Mixed group gets up to a 0.78 F1 score in the Spanish language.

When we look at the results of SemEval 2018 task 1EC, we notice a significant difference when compared to the SST5 dataset. It is surprising to see that performance improves in some groups, such as Slavic, by as much as 20 percent in F1-score with a recall rate of 30 percent in the baseline. We can also see significant differences when compared to the LASER; the F1-score increased by nearly 12 percent, with recall 19 percent in the Slavic group. This anomaly, on the other hand, can be easily explained by looking at Table 13. When we train the model with multiple languages, we can apparently see an improvement in the model's ability to capture difficult emotions such as "surprise." It is activated, and on average, we received a 0.15 in F1-score for this emotion; we can conclude that this factor can now explain the vast difference in results between SST5 and SemEval 2018 task 1EC results.

Language	Label	Germanic			Romance			Slavic			Mix		
		Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1	Pre	Rec	F1
English	Anger	0.70	0.65	0.67	0.70	0.67	0.69	0.70	0.63	0.67	0.71	0.67	0.69
English	Joy	0.82	0.78	**0.80**	0.82	0.78	0.80	0.81	0.77	0.79	0.83	0.78	0.80
English	Disgust	0.67	0.64	0.65	0.66	0.66	0.66	0.65	0.64	0.65	0.67	0.63	0.65
English	Surprise	0.28	0.12	0.14	0.19	0.14	0.16	0.23	0.15	0.18	0.23	0.09	0.13
Dutch	Anger	0.71	0.66	0.68	0.73	0.65	0.69	0.72	0.66	0.69	0.72	0.67	0.69
Dutch	Joy	0.81	0.77	0.79	0.81	0.78	**0.80**	0.81	0.78	0.79	0.82	0.77	**0.80**
Dutch	Disgust	0.68	0.64	0.66	0.69	0.63	0.66	0.67	0.64	0.66	0.68	0.63	0.66
Dutch	Surprise	0.24	0.14	0.17	0.20	0.09	0.13	0.25	0.14	0.18	0.33	0.10	0.15
German	Anger	0.72	0.67	0.69	0.72	0.66	0.69	0.72	0.66	0.68	0.73	0.66	0.69
German	Joy	0.82	0.75	0.78	0.81	0.76	0.78	0.81	0.76	0.78	0.83	0.75	0.79
German	Disgust	0.68	0.64	0.66	0.69	0.62	0.65	0.68	0.65	0.66	0.69	0.63	0.66
German	Surprise	0.20	0.10	0.13	0.23	0.12	0.16	0.25	0.14	0.18	0.33	0.11	0.16
Swedish	Anger	0.71	0.63	0.67	0.73	0.63	0.68	0.70	0.63	0.67	0.73	0.64	0.68
Swedish	Joy	0.81	0.77	0.79	0.80	0.79	0.79	0.80	0.77	0.78	0.82	0.76	0.79
Swedish	Disgust	0.68	0.63	0.66	0.69	0.61	0.65	0.68	0.65	0.66	0.68	0.63	0.66
Swedish	Surprise	0.22	0.11	0.15	0.25	0.11	0.15	0.28	0.14	0.18	0.36	0.09	0.15
Russian	Anger	0.71	0.68	0.69	0.72	0.66	0.69	0.72	0.68	**0.70**	0.72	0.66	0.69
Russian	Joy	0.83	0.77	0.79	0.81	0.77	0.79	0.82	0.76	0.79	0.83	0.76	0.79
Russian	Disgust	0.67	0.65	0.66	0.69	0.64	0.66	0.67	0.65	0.66	0.68	0.65	0.67
Russian	Surprise	0.24	0.12	0.16	0.24	0.12	0.16	0.27	0.14	0.18	0.31	0.10	0.15
Slovak	Anger	0.71	0.67	0.69	0.72	0.65	0.68	0.71	0.65	0.68	0.72	0.65	0.68
Slovak	Joy	0.81	0.77	0.79	0.8	0.78	0.79	0.81	0.77	0.79	0.83	0.76	0.79
Slovak	Disgust	0.66	0.61	0.64	0.69	0.60	0.64	0.67	0.63	0.65	0.69	0.62	0.65
Slovak	Surprise	0.19	0.09	0.12	0.25	0.09	0.14	0.23	0.12	0.16	0.32	0.08	0.13
Polish	Anger	0.71	0.67	0.69	0.73	0.66	0.69	0.70	0.66	0.68	0.73	0.66	**0.70**
Polish	Joy	0.82	0.77	0.79	0.81	0.78	0.79	0.81	0.77	0.79	0.82	0.75	0.79
Polish	Disgust	0.68	0.64	0.66	0.68	0.61	0.64	0.68	0.65	0.66	0.68	0.61	0.64
Polish	Surprise	0.26	0.13	0.17	0.28	0.12	0.17	0.27	0.15	0.19	0.29	0.09	0.14
Czech	Anger	0.71	0.66	0.68	0.72	0.64	0.68	0.72	0.66	0.69	0.73	0.65	0.69
Czech	Joy	0.81	0.76	0.79	0.80	0.78	0.79	0.81	0.78	0.79	0.82	0.76	0.78
Czech	Disgust	0.69	0.62	0.65	0.70	0.60	0.65	0.68	0.63	0.65	0.70	0.61	0.65
Czech	Surprise	0.23	0.09	0.13	0.25	0.08	0.12	0.30	0.14	**0.19**	0.31	0.07	0.11
Italian	Anger	0.69	0.67	0.68	0.70	0.68	0.69	0.70	0.67	0.68	0.71	0.68	0.69
Italian	Joy	0.81	0.77	0.79	0.81	0.77	0.79	0.81	0.77	0.79	0.82	0.76	0.79
Italian	Disgust	0.66	0.67	0.66	0.68	0.66	0.67	0.66	0.67	**0.67**	0.67	0.68	**0.67**
Italian	Surprise	0.19	0.12	0.15	0.23	0.14	0.17	0.25	0.15	0.19	0.28	0.10	0.15
French	Anger	0.70	0.67	0.69	0.72	0.66	0.69	0.71	0.64	0.67	0.71	0.66	0.69
French	Joy	0.80	0.78	0.79	0.82	0.77	0.79	0.80	0.78	0.79	0.81	0.77	0.79
French	Disgust	0.65	0.68	0.67	0.68	0.66	0.67	0.65	0.67	0.66	0.65	0.70	0.67
French	Surprise	0.24	0.14	0.17	0.23	0.11	0.15	0.29	0.15	0.20	0.23	0.11	0.15
Portuguese	Anger	0.71	0.67	0.69	0.7	0.67	0.69	0.71	0.66	0.69	0.72	0.68	**0.70**
Portuguese	Joy	0.81	0.76	0.79	0.81	0.76	0.78	0.81	0.75	0.78	0.83	0.75	0.79
Portuguese	Disgust	0.66	0.63	0.64	0.67	0.65	0.66	0.65	0.65	0.65	0.66	0.64	0.65
Portuguese	Surprise	0.20	0.12	0.15	0.19	0.12	0.15	0.25	0.14	0.17	0.30	0.09	0.14
Spanish	Anger	0.70	0.66	0.68	0.70	0.66	0.68	0.7	0.63	0.66	0.71	0.66	0.68
Spanish	Joy	0.82	0.76	0.79	0.82	0.76	0.79	0.82	0.76	0.79	0.83	0.76	**0.80**
Spanish	Disgust	0.66	0.64	0.65	0.67	0.65	0.66	0.67	0.65	0.66	0.67	0.64	0.65
Spanish	Surprise	0.22	0.12	0.15	0.2	0.12	0.15	0.23	0.13	0.17	0.31	0.10	0.15

Table 13: Micro averaged F1 scores on the SemEval 2018 task 1EC test ran once when SVM classifier with Gaussian kernel trained emEval 2018 task 1EC family groups training dataset using LASER embedding.

4.4 LaBSE

Language-agnostic BERT sentence embedding is a method that uses masked language model (MLM) and translation language model (TLM) techniques to achieve the results. One of the main differences except architecture compared to the LASER is that it predicts the next masked word without relying on the parallel data. It allows the self-attention layer to draw on information from both the left and right contexts of a word to make its prediction. Then, we also used different pre-processing techniques, such as the original Multilingual Preprocessor[12] after cleaning noise from the text.

For this experiment, we used three different implementations of the LaBSE model, including the official approach[13]. All of them retrieved the same results at the end. However, unfortunately, we could not reach the same results as the LASER model does. See Table 14, which shows that, on average, the model provides a 0.35 F1 score. It is interesting to see that the precision is very high, on average 0.95, which is much higher than it appeared LASER results. However, it is easily explained by the fact that a high recall rate was associated with the label "neutral." In general, it is hard to explain why we got these results by using LaBSE embeddings, and we want to leave it for future work due to time limits.

Language	Label	Precision	Recall	F1
English	Positive	**0.98**	**0.22**	**0.35**
English	Negative	0.94	**0.12**	**0.21**
Dutch	Positive	0.95	0.21	0.34
Dutch	Negative	**0.97**	0.07	0.13
German	Positive	0.97	0.20	0.34
German	Negative	0.90	0.09	0.16
Swedish	Positive	0.96	0.20	0.33
Swedish	Negative	0.87	0.09	0.16

Table 14: Micro averaged F1 scores on the SST5 test ran once when SVM classifier with Gaussian kernel trained SST5 training dataset by using LaBSE embedding.

[12]https://tfhub.dev/google/universal-sentence-encoder-cmlm/multilingual-preprocess/2
[13]https://tfhub.dev/google/LaBSE/2

5 Conclusion and Discussion

In this study, we ran an experiment with existing language-independent methodologies and analyzed their performance to see if they are truly language-independent. We chose two cutting-edge methods for this task: LASER and LaBSE. We also tested the performance of FastText word embeddings trained on translated datasets. Most importantly, we are able to answer the study's research questions.

***Q1**: How can language-independent word representations be used for more complex linguistic tasks such as sentiment analysis and emotion classification?*

As an example of how language-independent word representation can be used in sentiment analysis or emotion classification, we demonstrated several different approaches. The most straightforward method is to use FastText word embedding, which extracts features from datasets. However, the main disadvantage of this technique is that it uses a pre-trained model for each language separately, and it requires an average of 1.7 GB of memory on a single computer. For instance, to train models for 12 languages, we had to use 20.4 GB of pre-trained models. It means that we will be unable to easily and inexpensively expand our classifier to include more languages.

On the other hand, we made use of sentence representations that were not dependent on the language being used, such as LASER and LaBSE. Both of these techniques provide a universal pre-trained model that can be used to learn a variety of languages in a single session. Overall, we can train a classifier with only one encoded embedded representation and predict the sentence labels in other languages. In practice, this means that we can only use datasets that are in a single language and then use machine learning model knowledge to classify datasets from different languages. Nevertheless, the number of languages supported by a particular pre-trained model is also a consideration. LASER, for example, supports 93 different languages, while LaBSE supports 109 different languages. We can also train a classifier with multiple language datasets in order to improve its performance, as we did in the case of the LASER-FG. Despite the fact that this approach did not achieve significant improvement in F1 score with 3 percent in the SST5 dataset. It remained almost the same as the classifier trained only with an English encoded dataset. However, we can still see good improvement in recall, which is essential to note. A company such as ZyLAB, for example, examines a large number of documents, making it critical to have fewer errors in classifier prediction and a high recall rate, which means this approach still an excellent candidate to consider. Furthermore, it demonstrated that it was capable of dealing with emotions such as "surprise" and that it could be applied in the same situation when a classifier for similar classes was required.

***Q2** What are the results of the classifier trained with language-independent representations for selected twelve languages?*

With a Gaussian kernel and several language-independent representations, we train our SVM classifier to recognize patterns in data. Despite the fact that the datasets had been translated, the results of FastText word embedding in Table 4 and Table 5 showed promising results. For example, the original dataset in English yielded results that were nearly identical to those obtained in other languages, with a small amount of variation. While this deviation may seem

Language	FastText	LASER	LASER-FG	LaBSE
Czech	0.55	0.61	0.64	0.21
Dutch	0.58	0.62	0.65	0.23
English	0.61	0.61	0.64	0.28
French	0.56	0.59	0.65	0.23
German	0.57	0.62	0.65	0.24
Italian	0.57	0.61	0.65	0.23
Polish	0.56	0.62	**0.66**	0.21
Portuguese	0.57	0.61	0.65	0.22
Russia	0.57	0.62	0.65	0.22
Slovak	0.54	0.62	0.65	0.23
Spanish	0.58	0.60	0.65	0.24
Swedish	0.56	0.61	0.65	0.24

Table 15: Averaged F1 scores of FastText, LASER, LASER-FG and LaBSE results.

significant in theory, in practice, it may not be nearly as significant as the fact that it can be used in other languages.

When compared to the baseline, the LASER approach produces excellent results. During SST5 and SemEval 2018 task 1EC, it improved by 7 percent and 7.9 percent, respectively. Furthermore, it does not necessitate the translation of datasets or the use of multiple pre-trained models, and it eliminates the need for manual labor.

As shown in Table 11, the LASER-FG approach achieved even better results in SemEval 2018 1EC, achieving up to a 20% improvement over the baseline and approximately a 12 percent improvement over the LASER approach. The fact that it is a Slavic group is the most surprising aspect. However, when compared to the LASER dataset, the results of the SST5 dataset show a 3 percent improvement. Before employing this strategy, it is necessary to consider the context in which it will be used. Because, in practice, it may not be worthwhile to train a classifier with three languages for a 3 percent increase in classification accuracy. However, it may be worthwhile to train classifiers in order to improve recall or to capture some classes that are difficult to capture in a textual format.

The results of LaBSE are shown in Table 14, and it is clear that it was unable to achieve results that were comparable to those of LASER. We are unable to determine the cause of this behavior because we rule out differences in implementation and maintain it at the same level as LASER.

It is clear that when compared to the other techniques, LASER-FG yields more favorable results in the majority of cases, as shown in Table 15. The average F1 score is 0.65, which is the highest score that provides more quality results. To consider is the perspective of time and memory, which we do have available to us. More resources are required than with LASER to accomplish this task. It means that the LASER model is the quickest and easiest to use while still providing good performance. The objective of application plays a vital role, based on the needs of the company can be used LASER or LASER-FG.

Q3 *Do the linguistic properties have an effect on the classifier?*

We came to the conclusion that language properties have no effect on the classifier based on the results shown in Tables 12 and 13. Only a few insignificant points, for example, the Romance group performing less well in the English language when compared to the other groups, but still achieving acceptable results in the end. It is possible that the fact that our datasets were translated has something to do with the fact that the language properties have little influence on the classifier. As previously stated, Google Translation and LASER both employ the encode-decode architecture of neural machine translation in their operations. It is possible that this is the reason why it is not possible to transfer the unique properties and knowledge of these datasets that are the same.

Q3.1 *What type of linguistic properties is responsible for specific errors?*

As our LASER approach performed well in all languages. As a result, we could not find any pattern of errors did the classifier. However, in general, When it comes to classification in a multilingual context, there are several mistakes that might occur. One of the reasons might be the problem of translation. It is possible that the translation will provide words that do not have standard translations but rather have vulgar or colloquial meanings as a result of the translation [89], [90]. Ambiguity can also occur after a translation has been completed; for example, the word "remarks" in the English language may be incorrectly translated into the Spanish language to mean "observation," resulting in ambiguity [89]. Two words combined to convey a specific meaning can appear to be misspelled even when the translation is correct, as in the case of the Spanish word "legislaciones nacionales," which is translated from "national rulings" and appears to be misspelled [89]. In addition, fluency errors can appear as well in the sense of lexicon or grammar[90]. Furthermore, researchers proved that translation performance depending on sentence length and more number of words has more translation results could be degraded[91], [92].

Q3.2 *Is it possible to address classifier errors by using pre-processing or post-processing methods?*

When performing multilingual analysis, it is critical to use pre-processing techniques. There are a variety of techniques available and, consequently, differences in the effectiveness of the various techniques. We consider some of them. For instance, stemming improves the accuracy of classification during the classification process. It involves providing the root form or meaning of a word, for instance, removing suffixes. However, in some instances, it can result in the loss of emotional significance [93].

Similarly, lemmatization can be used in classification by finding the root form of the word [93]. This technique is essential as it improves accuracy levels. Lemmatization is more effective in ensuring that the actual word is found. Another technique that is employed is the removal of punctuation. It should be removed or replaced with words that provide better descriptions. Additionally, negative words or shortened forms of words can be substituted in order to improve accuracy. It entails filtering out specific words from the text.

We could generalize the texts to a kind of formal language, which would be a step forward [94], for instance, when you deal with tweets. Then, automatic spelling corrections can also

be considered in such context [95].

Meemi is a post-processing method that is used in cross-lingual classification. This technique, which takes into account the linear transformation of each language. As a result, it is possible to analyze multilingual information and use it to more effectively create more accurate embeddings[89]. In addition, the different strategies of whitening can also be used to arrive at the initial co-variances [89].

We can recommend the development of a production-ready language-independent model after taking all of the factors into consideration. We can see that LASER trained with a single language improves by 7 percent when compared to the Baseline with FastText, and LASER-FG improves by more than 10 percent in the SST5 dataset and by approximately 21 percent in the SemEval 2018 task 1EC dataset, respectively. However, it is possible that the results do not appear to be significant enough to be used for real-world tasks such as those in production; software companies such as ZyLAB, for example, may neglect accuracy over other parameters such as memory or the cost of model maintenance.

The LASER, for example, provides high-speed performance, with the ability to process up to 2,000 sentences per second when the GPU machines are used to process. The LASER pre-trained model, which is approximately 200 MB in size, costs nine times less than the FastText one-language pre-trained model. If there are parallel corpora available, it is also possible to create a custom pre-trained model using the LASER architecture. For example, ZyLAB is a company that specializes in eDiscovery in the legal industry. It means that they need models that recognize specific terms of the law field. A multilingual pre-trained model focusing on law terms might improve the accuracy of provided eDiscovery services. In addition, LASER uses a few external dependencies on the PyTorch implementation of the sentence encoder, which is also an essential aspect. In terms of money, it also demonstrates cost-effectiveness when compared to services such as Google Translator, where companies must pay per 10000 characters while using their services.

Besides providing all of the previously mentioned benefits, it also includes additional NLP tasks that may be beneficial for the ZyLAB. For example, it can be used to mine parallel data in large collections of monolingual data, or in multilingual similarity search, or in cross-lingual document classification and cross-lingual natural language inference.

Finally, multilingual joint training can be beneficial for languages with limited resources because it allows them to learn more than one language at the same time. When more than one language is used in a single sentence, the model allows for that. As more languages are added, the system's performance improves as it learns to recognize the characteristics of different language families and as it becomes more efficient. Furthermore, As long as all of these languages are encoded by the same BiLSTM encoder, there is no need to specify which language is being input. As far as we can tell from our experience, the sentence encoder also supports code-switching, which means that the same sentences can contain words in multiple languages at the same time. The authors [17] also have evidence that the encoder can generalize to other languages that have not been seen during training but are members of a language family covered by other languages, which we are currently investigating.

6 Future work

Based on the results of the experiments we conducted, we came to the conclusion that it may be beneficial to use more datasets with original languages and train the classifier in order to improve accuracy. Because we used translated datasets, it is possible that the original datasets contain more comprehensive information about language properties. Then we will be able to see clearly how the language properties influence the classifier. Aside from that, it might be beneficial to combine datasets from different contexts. For example, data collected from a tweeter concatenate with a customer review dataset. It has the potential to either improve or degrade the outcome of the classifier. However, it is a fascinating subject to look into, regardless of the outcome. More pre-processing techniques, such as removing negation from the text, should be attempted as a counter-example.

7 References

[1] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *arXiv preprint arXiv:1310.4546*, 2013.

[2] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[3] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, PMLR, 2014, pp. 1188–1196.

[4] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, A. Torralba, R. Urtasun, and S. Fidler, "Skip-thought vectors," *arXiv preprint arXiv:1506.06726*, 2015.

[5] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[6] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, *et al.*, "Universal sentence encoder for english," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 169–174.

[7] S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," *arXiv preprint arXiv:1804.00079*, 2018.

[8] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

[9] W. Ammar, G. Mulcaire, Y. Tsvetkov, G. Lample, C. Dyer, and N. A. Smith, "Massively multilingual word embeddings," *arXiv preprint arXiv:1602.01925*, 2016.

[10] S. Ruder, I. Vulić, and A. Søgaard, "A survey of cross-lingual word embedding models," *Journal of Artificial Intelligence Research*, vol. 65, pp. 569–631, 2019.

[11] M. Artetxe, G. Labaka, and E. Agirre, "Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations," in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[12] ——, "A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings," *arXiv preprint arXiv:1805.06297*, 2018.

[13] H. Schwenk and M. Douze, "Learning joint multilingual sentence representations with neural machine translation," *arXiv preprint arXiv:1704.04154*, 2017.

[14] M. Johnson, M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, *et al.*, "Google's multilingual neural machine translation system: Enabling zero-shot translation," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 339–351, 2017.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[17] M. Artetxe and H. Schwenk, "Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 597–610, 2019.

[18] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, and W. Wang, "Language-agnostic bert sentence embedding," *arXiv preprint arXiv:2007.01852*, 2020.

[19] S. Wu and M. Dredze, "Beto, bentz, becas: The surprising cross-lingual effectiveness of bert," *arXiv preprint arXiv:1904.09077*, 2019.

[20] E. M. Bender, "Linguistically naıve!= language independent: Why nlp needs linguistic typology," in *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, 2009, pp. 26–32.

[21] T. Naseem, R. Barzilay, and A. Globerson, "Selective sharing for multilingual dependency parsing," The Association for Computational Linguistics, 2012.

[22] D. W. Oard, J. R. Baron, B. Hedin, D. D. Lewis, and S. Tomlinson, "Evaluation of information retrieval for e-discovery," *Artificial Intelligence and Law*, vol. 18, no. 4, pp. 347–386, 2010.

[23] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, NIH Public Access, vol. 11, 2017, p. 269.

[24] I. Caswell and B. Liang, "Recent advances in google translate," *Google AI Blog*, 2020.

[25] H. Schwenk and X. Li, "A corpus for multilingual document classification in eight languages," *arXiv preprint arXiv:1805.09821*, 2018.

[26] S. Rosenthal, N. Farra, and P. Nakov, "Semeval-2017 task 4: Sentiment analysis in twitter," in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, 2017, pp. 502–518.

[27] S. Mohammad, F. Bravo-Marquez, M. Salameh, and S. Kiritchenko, "Semeval-2018 task 1: Affect in tweets," in *Proceedings of the 12th international workshop on semantic evaluation*, 2018, pp. 1–17.

[28] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[29] H. Huang, H. Wang, and D. Jin, "A low-cost named entity recognition research based on active learning," *Scientific Programming*, vol. 2018, 2018.

[30] M. Devika, C. Sunitha, and A. Ganesh, "Sentiment analysis: A comparative study on different approaches," *Procedia Computer Science*, vol. 87, pp. 44–49, 2016.

[31] S. Clark and S. Pulman, "Combining symbolic and distributional models of meaning," 2007.

[32] J. Mitchell and M. Lapata, "Vector-based models of semantic composition," in *proceedings of ACL-08: HLT*, 2008, pp. 236–244.

[33] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 43–52, 2010.

[34] S. Qaiser and R. Ali, "Text mining: Use of tf-idf to examine the relevance of words to documents," *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018.

[35] D. M. El-Din, "Enhancement bag-of-words model for solving the challenges of sentiment analysis," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 1, 2016.

[36] D. Ho, A. S. Shkolnik, N. J. Ferraro, B. A. Rizkin, and R. L. Hartman, "Using word embeddings in abstracts to accelerate metallocene catalysis polymerization research," *Computers & Chemical Engineering*, vol. 141, p. 107 026, 2020.

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

[38] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[39] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum, "Efficient non-parametric estimation of multiple embeddings per word in vector space," *arXiv preprint arXiv:1504.06654*, 2015.

[40] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759*, 2016.

[41] B. Kuyumcu, C. Aksakalli, and S. Delil, "An automated new approach in fast text classification (fasttext) a case study for turkish text classification without preprocessing," in *Proceedings of the 2019 3rd International Conference on Natural Language Processing and Information Retrieval*, 2019, pp. 1–4.

[42] A. Agibetov, K. Blagec, H. Xu, and M. Samwald, "Fast and scalable neural embedding models for biomedical sentence classification," *BMC bioinformatics*, vol. 19, no. 1, pp. 1–9, 2018.

[43] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[44] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[45] K. Singla, D. Can, and S. Narayanan, "A multi-task approach to learning multilingual representations," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2018, pp. 214–220.

[46] H. Pham, M.-T. Luong, and C. D. Manning, "Learning distributed representations for multilingual text sequences," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 2015, pp. 88–94.

[47] O. Adams, A. Makarucha, G. Neubig, S. Bird, and T. Cohn, "Cross-lingual word embeddings for low-resource language modeling," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 937–947.

[48] Y. Doval, J. Camacho-Collados, L. Espinosa-Anke, and S. Schockaert, "Improving cross-lingual word embeddings by meeting in the middle," *arXiv preprint arXiv:1808.08780*, 2018.

[49] Y.-H. Lin, C.-Y. Chen, J. Lee, Z. Li, Y. Zhang, M. Xia, S. Rijhwani, J. He, Z. Zhang, X. Ma, *et al.*, "Choosing transfer languages for cross-lingual learning," *arXiv preprint arXiv:1905.12688*, 2019.

[50] J. C. B. Cruz and C. Cheng, "Evaluating language model finetuning techniques for low-resource languages," *arXiv preprint arXiv:1907.00409*, 2019.

[51] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *arXiv preprint arXiv:1409.3215*, 2014.

[52] A. Saha, M. M. Khapra, S. Chandar, J. Rajendran, and K. Cho, "A correlational encoder decoder architecture for pivot based sequence generation," *arXiv preprint arXiv:1606.04754*, 2016.

[53] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[54] R. Vázquez, A. Raganato, M. Creutz, and J. Tiedemann, "A systematic study of inner-attention-based sentence representations in multilingual neural machine translation," *Computational Linguistics*, vol. 46, no. 2, pp. 387–424, 2020.

[55] H. Hammarström, R. Forkel, and M. Haspelmath, "Glottolog 3.0," *Max Planck Institute for the Science of Human History*, 2017.

[56] E. M. Ponti, H. O'horan, Y. Berzak, I. Vulić, R. Reichart, T. Poibeau, E. Shutova, and A. Korhonen, "Modeling language variation and universals: A survey on typological linguistics for natural language processing," *Computational Linguistics*, vol. 45, no. 3, pp. 559–601, 2019.

[57] M. S. Dryer and M. Haspelmath, "The world atlas of language structures online. leipzig: Max planck institute for evolutionary anthropology," *Online: http://wals.info*, 2013.

[58] H. O'Horan, Y. Berzak, I. Vulić, R. Reichart, and A. Korhonen, "Survey on the use of typological information in natural language processing," *arXiv preprint arXiv:1610.03349*, 2016.

[59] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi, "Integrating and evaluating neural word embeddings in information retrieval," in *Proceedings of the 20th Australasian document computing symposium*, 2015, pp. 1–8.

[60] S. B. Cohen, D. Das, and N. A. Smith, "Unsupervised structure prediction with non-parallel multilingual guidance," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011, pp. 50–61.

[61] D. M. Gaddy, Y. Zhang, R. Barzilay, and T. S. Jaakkola, "Ten pairs to tag-multilingual pos tagging via coarse mapping between embeddings," Association for Computational Linguistics, 2016.

[62] A. Klementiev, I. Titov, and B. Bhattarai, "Inducing crosslingual distributed representations of words," in *Proceedings of COLING 2012*, 2012, pp. 1459–1474.

[63] V. N. Gudivada, D. Rao, and V. V. Raghavan, "Big data driven natural language processing research and applications," in *Handbook of Statistics*, vol. 33, Elsevier, 2015, pp. 203–238.

[64] H. Shi-lin, Z. Xiao-lin, and C. De-ren, "A semi-supervised learning method for product named entity recognition," *Journal of Beijing University of Posts and Telecommunications*, vol. 36, no. 2, p. 20, 2013.

[65] E. Cambria, D. Das, S. Bandyopadhyay, and A. Feraco, *A practical guide to sentiment analysis*. Springer, 2017.

[66] J. Zhou and J.-m. Ye, "Sentiment analysis in education research: A review of journal publications," *Interactive Learning Environments*, pp. 1–13, 2020.

[67] S. D'Mello and A. Graesser, "Dynamics of affective states during complex learning," *Learning and Instruction*, vol. 22, no. 2, pp. 145–157, 2012.

[68] L. Nahar, Z. Sultana, N. Iqbal, and A. Chowdhury, "Sentiment analysis and emotion extraction: A review of research paradigm," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, IEEE, 2019, pp. 1–8.

[69] G. Gautam and D. Yadav, "Sentiment analysis of twitter data using machine learning approaches and semantic analysis," in *2014 Seventh International Conference on Contemporary Computing (IC3)*, IEEE, 2014, pp. 437–442.

[70] Z. Shao, R. Chandramouli, K. Subbalakshmi, and C. T. Boyadjiev, "An analytical system for user emotion extraction, mental state modeling, and rating," *Expert Systems with Applications*, vol. 124, pp. 82–96, 2019.

[71] C. Argueta, F. H. Calderon, and Y.-S. Chen, "Multilingual emotion classifier using unsupervised pattern extraction from microblog data," *Intelligent Data Analysis*, vol. 20, no. 6, pp. 1477–1502, 2016.

[72] P. Gilbert, "The relationship of shame, social anxiety and depression: The role of the evaluation of social rank," *Clinical Psychology & Psychotherapy: An International Journal of Theory & Practice*, vol. 7, no. 3, pp. 174–189, 2000.

[73] D. Chatzakou, A. Vakali, and K. Kafetsios, "Detecting variation of emotions in online activities," *Expert Systems with Applications*, vol. 89, pp. 318–332, 2017.

[74] M. A. Azim and M. H. Bhuiyan, "Text to emotion extraction using supervised machine learning techniques," *Telkomnika*, vol. 16, no. 3, pp. 1394–1401, 2018.

[75] M. Gjoreski, H. Gjoreski, and A. Kulakov, "Machine learning approach for emotion recognition in speech," *Informatica*, vol. 38, no. 4, 2014.

[76] Q. V. Le and M. Schuster, "A neural network for machine translation, at production scale," *Google research blog*, vol. 27, 2016.

[77] M. Z. Asghar, A. Khan, S. Ahmad, and F. M. Kundi, "A review of feature extraction in sentiment analysis," *Journal of Basic and Applied Scientific Research*, vol. 4, no. 3, pp. 181–186, 2014.

[78] S. Vychegzhanin and E. Kotelnikov, "Comparison of named entity recognition tools applied to news articles," in *2019 Ivannikov Ispras Open Conference (ISPRAS)*, IEEE, 2019, pp. 72–77.

[79] R. Jongeling, S. Datta, and A. Serebrenik, "Choosing your weapons: On sentiment analysis tools for software engineering research," in *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, IEEE, 2015, pp. 531–535.

[80] U. Malik, *Python for nlp: Tokenization, stemming, and lemmatization with spacy library*. [Online]. Available: `https://stackabuse.com/python-for-nlp-tokenization-stemming-and-lemmatization-with-spacy-library/`..

[81] Y. Yang, G. H. Abrego, S. Yuan, M. Guo, Q. Shen, D. Cer, Y.-H. Sung, B. Strope, and R. Kurzweil, "Improving multilingual sentence embedding using bi-directional dual encoder with additive margin softmax," *arXiv preprint arXiv:1902.08564*, 2019.

[82] M. Guo, Q. Shen, Y. Yang, H. Ge, D. Cer, G. H. Abrego, K. Stevens, N. Constant, Y.-H. Sung, B. Strope, *et al.*, "Effective parallel corpus mining using bilingual sentence embeddings," *arXiv preprint arXiv:1807.11906*, 2018.

[83] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.

[84] L. Gong, D. He, Z. Li, T. Qin, L. Wang, and T. Liu, "Efficient training of bert by progressively stacking," in *International Conference on Machine Learning*, PMLR, 2019, pp. 2337–2346.

[85] H. William, S. Teukolsky, W. Vetterling, and B. Flannery, "What is a support vector machine," *Nat Biotechnol*, vol. 24, pp. 1565–1567, 2006.

[86] Q. Wu and D.-X. Zhou, "Analysis of support vector machine classification.," *Journal of Computational Analysis & Applications*, vol. 8, no. 2, 2006.

[87] N. Tatbul, T. J. Lee, S. Zdonik, M. Alam, and J. Gottschlich, "Precision and recall for time series," *arXiv preprint arXiv:1803.03639*, 2018.

[88] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual bert?" *arXiv preprint arXiv:1906.01502*, 2019.

[89] Y. Doval, J. Camacho-Collados, L. E. Anke, and S. Schockaert, "Meemi: A simple method for post-processing cross-lingual word embeddings," *arXiv preprint arXiv:1910.07221*, 2019.

[90] L. Van Brussel, A. Tezcan, and L. Macken, "A fine-grained error analysis of nmt, smt and rbmt output for english-to-dutch," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[91] L. Bentivogli, A. Bisazza, M. Cettolo, and M. Federico, "Neural versus phrase-based machine translation quality: A case study," *arXiv preprint arXiv:1608.04631*, 2016.

[92] A. Toral and V. M. Sánchez-Cartagena, "A multifaceted evaluation of neural versus phrase-based machine translation for 9 language directions," *arXiv preprint arXiv:1701.02901*, 2017.

[93] N. Babanejad, A. Agrawal, A. An, and M. Papagelis, "A comprehensive analysis of preprocessing for word representation learning in affective tasks," in *Proceedings of the 58th annual meeting of the association for computational linguistics*, 2020, pp. 5799–5810.

[94] W. De Smet and M.-F. Moens, "Generating a topic hierarchy from dialect texts," in *18th International Workshop on Database and Expert Systems Applications (DEXA 2007)*, IEEE, 2007, pp. 249–253.

[95] T. Mullen and R. Malouf, "A preliminary investigation into sentiment analysis of informal political discourse.," in *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, 2006, pp. 159–162.