Universität Bielefeld
Fakultät für Linguistik und Literaturwissenschaft

Masterarbeit
im Studiengang "Linguistik: Kommunikation, Kognition und Sprachtechnologie"

zum Thema:

Exploring Crosslingual Word Embeddings for Semantic Classification in Text and Dialogue

Vorgelegt von
Oleksandra Vsesviatska

Erstgutachterin: Dr. Sina Zarrieß
Zweitgutachter: Prof. Dr. David Schlangen

Bielefeld, 09.07.2019

Abstract

Current approaches to learning crosslingual word emebeddings provide a decent performance when based on a big amount of parallel data. Considering the fact, that most of the languages are under-resourced and lack structured lexical materials, it makes it difficult to implement them into such methods, and, respectively, into any human language technologies.

In this thesis we explore whether crosslingual mapping between two sets of monolingual word embeddings obtained separately is strong enough to present competitive results on semantic classification tasks. Our experiment involves learning crosslingual transfer between German and French word vectors based on the combination of adversarial approach and the Procrustes algorithm. We evaluate embeddings on topic classification, sentiment analysis and humour detection tasks. We use a German subset of a multilingual data set for training, and a French subset for testing our models.

Results across German and French languages prove that word vectors mapped into a shared vector space are able to obtain and transfer semantic information from one language to another successfully. We also show that crosslingual mapping does not weaken the monolingual connections between words in one language.

Contents

Chapter 1

Introduction

1.1 Motivation

The active development of the field of Natural Language Processing (NLP), and in particular of its subfield Text Classification is largely related to an increase of the Internet usage growth (Sebastiani, 2002) (as of today, more than 50% of the world's population uses the Internet[1]). According to James (2016) over the last two years people produced 90% of all the data existed, and 2,5 quintillion bytes of data is created every day. In order to benefit from it, to foresee the coming changes and to make more informative decisions, both individuals and organizations are in need to radically automate and improve these information flows.

"Marketers have always needed to monitor media for information related to their brands — whether it's for public relations activities, fraud violations, or competitive intelligence. But fragmenting media and changing consumer behaviour have crippled traditional monitoring methods. Technorati estimates that 75,000 new blogs are created daily, along with 1.2 million new posts each day, many discussing consumer opinions on products and services. Tactics [of the traditional sort] such as clipping services, field agents, and ad hoc research simply can't keep pace."

(Kim, 2006)

[1]https://www.internetworldstats.com/stats.htm

Text Classification is the task of assigning a Boolean value to each pair $(d_j, c_k) \in D \times C$, where D is a collection of textual documents and $C = \{c_1, c_2, ..., c_{|C|}\}$ is a set of predefined categories. Sebastiani (2002) defines the task as approximation of the unknown target function $\phi : D \times C \rightarrow \{0, 1\}$ (that points out how documents have to be classified) by means of a classifier $K : D \times C \rightarrow \{0, 1\}$, such that ϕ and K match each other as much as possible. 1 denotes the decision of the classifier to assign d_j to c_k, while 0 means that according to the classifier d_j does not belong to c_k.

Before being classified text documents are converted into numeric vectors. By far, the most common transformation is the Bag-of-Words (BoW) model (e.g. McCallum et al. (1998), Joachims (1998)). It uses a binary vector of each word in the document (a one-hot encoding) to build a feature vector, in which every entry corresponds to the number of times the document contains a specific word. In spite of relatively satisfactory accuracy in categorizing documents, BoW model has significant limitations.

1. You look wonderful today
2. You look great today

Examples illustrated above provide the same semantic information. But visualizing a one-hot vector for each word in the vocabulary demonstrates that any dimension has no projection along the other ones. This means "wonderful" and "great" are in the same relation as "wonderful" and "look".

vocabulary= [you, look, wonderful, great, today]

you = [1,0,0,0,0], look=[0,1,0,0,0], wonderful=[0,0,1,0,0], great=[0,0,0,1,0], today=[0,0,0,0,1]

Yet representation of words or sequences with vectors reflecting their semantics is often essential. If two closely related words can be represented by close vectors, then such representations can then be effectively used for most of NLP tasks such as information retrieval, classification, text categorization or summarization, sentiment analysis, named entities recognition, solving word sense disambiguation, etc.

The distributional hypothesis, according to which, words that appear in similar contexts, have similar meanings, was presented in the 1950s (Firth, 1957; James, 2016). But a major breakthrough towards its practical confirmation occurred only in 2013, when Mikolov et al. put forward an efficient algorithm for training word embeddings (Mikolov et al., 2013c). The models that can obtain the meaning attained the extraordinary popularity in NLP community and served as inspiration for academics to arrive at many new ideas (e.g. Pennington et al. (2014); Gouws et al. (2015); Bojanowski et al. (2017)), one of which was to enable semantics across languages.

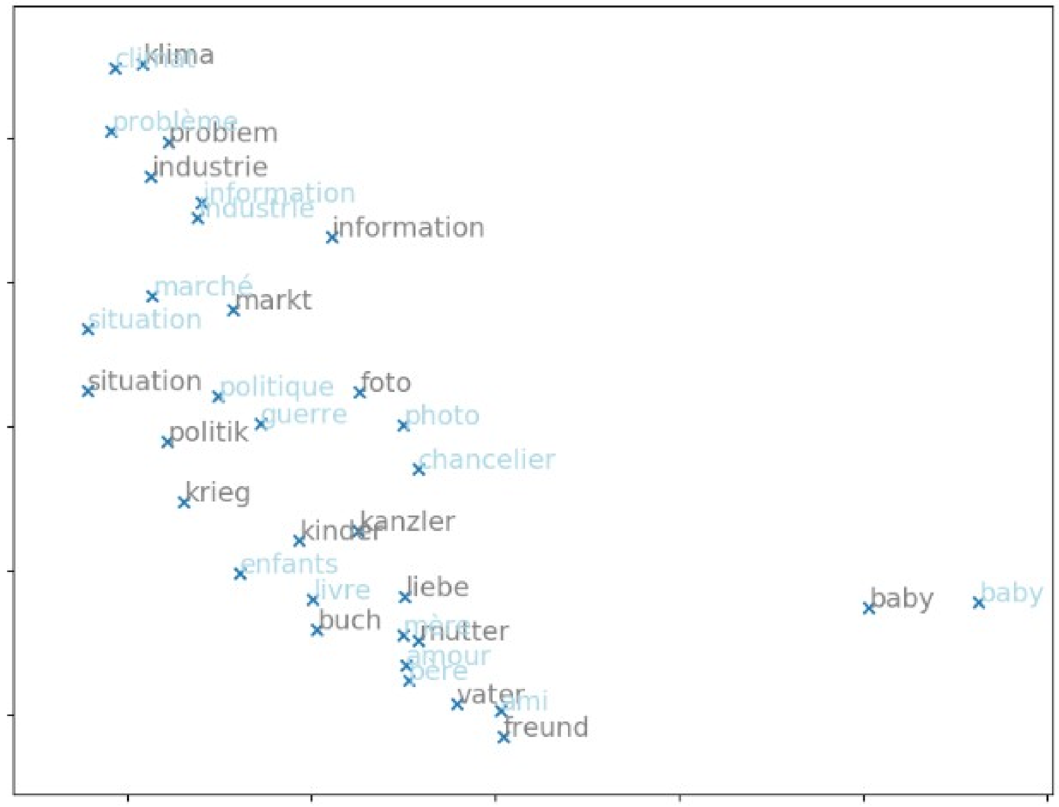


Figure 1.1: A shared embedding space between German and French, inspired by (Luong et al., 2015)

Most of the newest NLP and speech recognition tasks require an enormous amount of transcribed and annotated data. However, the reality suffers from the fact that there is only one language in the world that enjoys the large repository of available resources – English (Klementiev et al., 2012). The collection, annotation or phonemic transcription of data (e.g. for languages with no official writing system) require time and spending, so the data available is usually scarce or unannotated.

Recently there have been efforts on leveraging the power of word embeddings towards computation of crosslingual semantics (e.g. Das and Petrov (2011); Chandar et al. (2014); Duong et al. (2016)). Such crosslingual learning allows to project parameters learned from data in one language onto data in another language. In particular, the task is to learn these multilingual word embeddings for words sharing inter-lingual embedding space (see Figure 1.1). The rising attention to crosslingual models and their applications has become a motivation for this thesis.

1.2 Research Goals and Research Questions

Despite a strong interest in the field, most of the works on exploring crosslingual mapping evaluate the performance on 'ideal' balanced data sets with English training data. The main goal of this thesis is to conduct a deeper investigation on several research questions:

- Does the quality of mapping within one pair of languages depend on which embedding set is chosen as a source set and aligned into a shared space? Namely, in the crosslingual alignment, will a German-French mapping differ from a French-German mapping?
- How does the crosslingual mapping influence the quality of word embeddings?
- Are word embeddings able to pick up semantically challenging relations? And to transfer them across languages?

Moreover, while writing the thesis, the need of a more structured, simple and clear overview of the motivation, existing methods, tasks and application of crosslingual word embeddings was learned. Thus, presenting the overall picture of literature analysis has become one of the contributions this thesis has to offer.

Chapter 2

Background

2.1 Distributional Semantics

Distributional semantics is exploring ways to determine the semantic similarity of words based on their distribution in a large corpus of texts. The most successful methods for learning distributive representation of words (e.g. Mikolov et al. (2013c), Bojanowski et al. (2017) , Pennington et al. (2014)) are based on the distributive hypothesis of Harris stating that words with similar contexts have similar meanings. "You shall know the word by the company it keeps." (Harris, 1954).

2.1.1 Types of Semantic Similarity

In computational linguistics, two words are called semantically close (semantically similar) if they have a common hyperonym (parent category). For example, words *rose* and *tulip* are close, because they both are *flowers* (Resnik, 1995). This type of relations between words is sometimes called taxonomical similarity (Turney and Pantel, 2010).

Semantic similarity is a special case of semantic relatedness of words (Budanitsky and Hirst, 2001). Words are semantically related if they are in relation of meronymy (part to whole relation: *tale* and *dog*), hyponymy (generic relation: *dog* and *animal*), synonymy (*dog* and *hound*), antonymy (*fast* and *slow*) (Turney and Pantel, 2010). Words that tend to co-occur frequently (*dog* and *bark*) also belong to this category (Chiarello et al., 1990).

In applications, it is important to be able to distinguish semantic similarity of words from other types of semantic relatedness. At the same time, the exact definition of semantic similarity may vary depending on the formulation of the applied problem. For example, in order to automate a call center in a bank, it is important to exclude antonyms from the concept of related words. At the same time for automatic keyword replenishment system antonyms may be considered semantically close.

In order to determine the type of word similarity it is useful to note that there are two fundamentally different types of word co-occurrence in a corpus (Turney and Pantel, 2010). If two words are often found in the text next to each other, they are called syntagmatic associates. For instance, *pet* and *dog.* If two words are interchangeable in the same contexts, they are called paradigmatic parallels. For example: *pet* and *feed* (both words occur in the context of the word *dog*). Syntagmatic associates are usually semantically associated (*cat* and *purr* are often neighbours) and are often different parts of speech; paradigmatic parallels are taxonomically similar (*hound* and *dog* have similar neighbours).

Research in cognitive sciences went a few steps further and suggested other terms and applications for semantic relatedness - attributional similarity and relational similarity (Gentner, 1983). But in this case, it is not individual words that are involved in the relation, but relation itself. For example, such pairs as *Germany: Berlin* and *France: Paris* will have high similarity. This type of similarity gained its research popularity in 2013 after a number of articles by Mikolov et al. were published (Mikolov et al. (2013c),Mikolov et al. (2013d)). They proposed to solve the semantic analogy by guessing the fourth word from three known, for example *Japan: sushi, Germany: ?* or *Germany: Berlin, France : ?*. Mikolov et al. developed a word2vec program showing that a complicated analogy problem can be solved by simply adding and subtracting vectors learned with a neural network.

Later on, Mikolov et al. noticed similarities of semantic relations across languages. For example, the query *Germany: Berlin, Russia : ?* tested on embeddings obtained with word2vec from English data, and the query *Deutschland: Berlin, Russland : ?* from German data are likely to have the same response word - *Moscow* and *Moskau.* They proposed the way to exploit such similarities in a shared space, so that the semantically related words *hound* and *dog* had similar vectors to *hund* (Mikolov et al., 2013b).

2.1.2 Monolingual Word Embeddings

The idea of crosslingual mapping was based and is still inspired by the success of monolingual word embeddings for NLP tasks. Thus, it makes sense to briefly introduce the motivation and fundamental principles for learning monolingual distributed representations.

Count-based Word Representations

The classical approach called Bag of Word (BoW) has been adopted to represent each of words V as a numerical V-dimensional vector of all zeros, except for the one in index position i (see example sentences 1 and 2 on page 2) for many decades. Despite its success in the 90s (e.g. McCallum et al. (1998), Joachims (1998)), BoW lacked the only interesting function of words: their meaning, since all the words are at the same distance from each other in the vector space - 1. The same problem applies to documents: the semantic similarity of two texts (two sets of words) is estimated by the number of matching words. This means that two texts in which there are few or no common words are considered semantically and thematically not close. In Figure 2.1 both sentences share one piece of information but are represented by orthogonal vectors in BoW model in the vector space.

Moreover, representing words just as discrete independent ids leads to the data sparsity, that usually means the requirement of more data for successful model training.

Inspired by the distributional hypothesis of Harris (1954), saying that words that occur in the same contexts are likely to have similar meanings, and Bag of words hypothesis of Salton et al. (1975), stating that the number of words in a document can reflect its relevance to the query, Furnas et al. (1983) suggests a new hypothesis, that statistical count of word usage can depict the meaning of words. This hypothesis becomes fundamental in a new approach for solving NLP tasks - Vector Space model of Semantics (VSM). VSMs use frequencies to represent units of text by a vector in a text frequency matrix[1]. For example, for the document similarity tasks the rows of a text matrix correspond to the words in the dictionary, and the columns correspond to documents. The elements of the matrix are count vectors n_{ud}, which show

[1]Same as Turney and Pantel (2010), we use term “text frequency matrix” as a general case for any frequency matrix such as a term-document, word-context, or pair-pattern matrix.

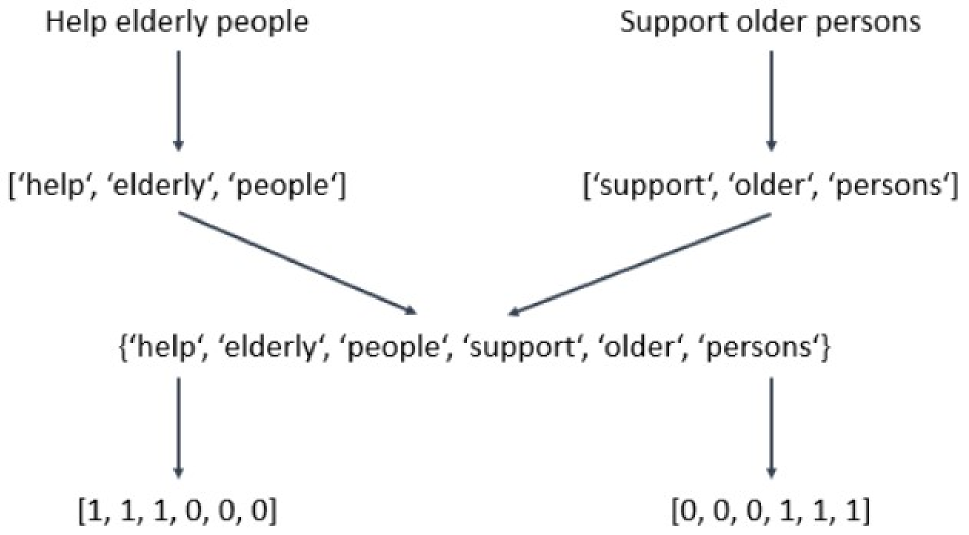
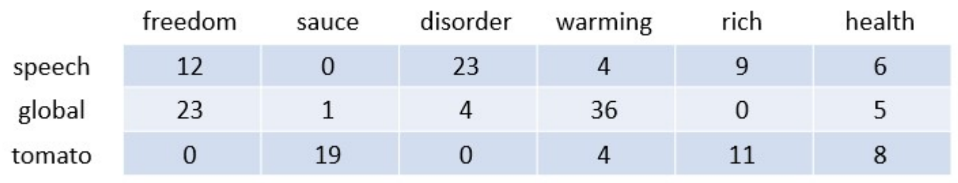

Help elderly people
Support older persons
['help', 'elderly', 'people']
['support', 'older', 'persons']
{'help', 'elderly', 'people', 'support', 'older', 'persons'}
[1, 1, 1, 0, 0, 0]
[0, 0, 0, 1, 1, 1]

Figure 2.1: The Bag of Words representation of two semantically related sentences

how many times the specific word u was used in a specific document d. The documents are similar if their vectors are similar (Furnas et al., 1983).



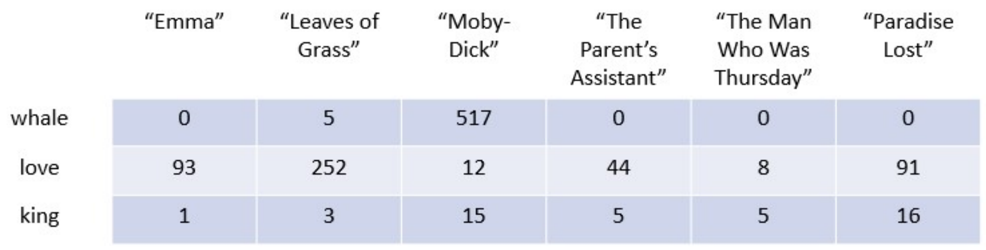
	freedom	sauce	disorder	warming	rich	health
speech	12	0	23	4	9	6
global	23	1	4	36	0	5
tomato	0	19	0	4	11	8

Table 2.1: Illustration of a word-context matrix inspired by Jurafsky and Martin (2014)

In the word-context matrix, the elements of the matrix are count vectors n_{uv}, which count the times the word u occurred in the context v. The concept of context can be defined in several ways. In the simplest case context is all words located no further than h positions from a given word in the text, i.e. in the window of a fixed radius. Thus, the considered frequency matrix will be a square symmetric co-occurrence matrix. This is also the most common case in the literature. In a more complex case, the syntactic structure of a sentence may be involved in the definition of context, for example, contexts can be considered:

- verbs in constructions such as subject-verb and verb-object (Hindle, 1990);
- one noun to the left and one noun to the right from the main noun in the sentence (Riloff and Shepherd, 1997)
- all adjectives in a noun phrase (Socher et al., 2007).

However, the work of Pantel et al. (2004) shows that in the case of a sufficiently large corpus, methods with no syntax analysis can also achieve comparable performance.



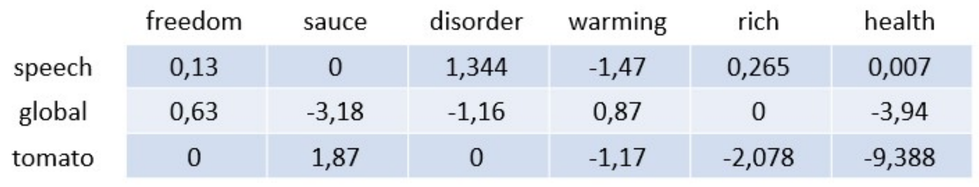
	"Emma"	"Leaves of Grass"	"Moby-Dick"	"The Parent's Assistant"	"The Man Who Was Thursday"	"Paradise Lost"
whale	0	5	517	0	0	0
love	93	252	12	44	8	91
king	1	3	15	5	5	16

Table 2.2: Illustration of a term-document matrix inspired by Jurafsky and Martin (2014)

An important problem of the raw word frequency matrix is the imbalance between rare and frequent items. For example, the row corresponding to the article ”the” will contain much higher count vectors than the row corresponding to the rare term ”symmetry”, however “the” might be not as discriminative. Moreover, the matrix may still be sparse, not dense. **Latent Semantic Analysis** (LSA) first introduced by Deerwester et al. (1990) has become one of the most popular methods for learning dense word representations. Every item of the sparse text frequency matrix A is replaced by its pointwise mutual information (PMI) score (Church and Hanks, 1990), creating a PMI matrix $pmi(A)$.

$$PMI(u,v) = \log \frac{p(u,v)}{p(u)p(v)}, \tag{2.1}$$

where $p(u, v)$ is the empirical probability of encountering two words in a fixed-width window, and $p(u)$ and $p(v)$ are empirical probabilities of meeting u and v in the corpus. The numerator shows how often two the target and the feature occur together, the denominator shows how often these words are expected to cooccur, taking to attention that they both occurred independently. If PMI is positive, the (u, v) pair is more likely to occur together than it would coincidentally (Jurafsky and Martin, 2014).



	freedom	sauce	disorder	warming	rich	health
speech	0,13	0	1,344	-1,47	0,265	0,007
global	0,63	-3,18	-1,16	0,87	0	-3,94
tomato	0	1,87	0	-1,17	-2,078	-9,388

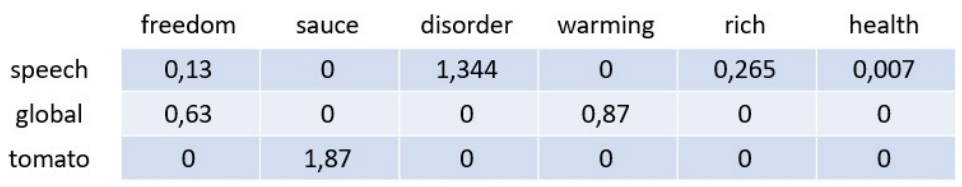
Table 2.3: Illustration of a PMI matrix inspired by Jurafsky and Martin (2014)

PMI successfully punishes too frequent words but has several drawbacks with the negative values. Firstly, words co-occur less often than we think by coincidence, secondly, negative values are unreliable without a large training corpus (Lin, 1998). In addition, the meanings are not defined for words that have never co-occurred. Bullinaria and Levy (2007) presented a simple solution of these problems - positive PMI (pPMI). pPMI replaces problematic negative values with 0.

$$pPMI(u, v) = \max(0, PMI(u, v)) \tag{2.2}$$

In the pPMI matrix $ppmi(A)$ each word is still represented by a long sparse vector of some values. Such a representation contains noise, and long sparse vectors may be ineffective as features for machine learning (less weights are easier to tune). That is why an important step is the transition to dense vectors in a space of lower dimension.

To reduce the data dimension LSA applies Singular Value Decomposition (SVD) (Deerwester et al., 1990). It is a separate important method for



	freedom	sauce	disorder	warming	rich	health
speech	0,13	0	1,344	0	0,265	0,007
global	0,63	0	0	0,87	0	0
tomato	0	1,87	0	0	0	0

Table 2.4: Illustration of a pPMI matrix inspired by Jurafsky and Martin (2014)

analyzing data, and the resulting matrix decomposition has a meaningful interpretation. Also, being a numerical method it can be used for Principal Component Analysis (PCA) (Pearson, 1901), (Hotelling, 1933), and with some reservations (Qiao, 2015) for Nonnegative Matrix Factorization (NMF) (Paatero and Tapper, 1994),(Lee and Seung, 2001), as well as for other analysis approaches to improve their performance. In addition, SVD does not have such inconvenient properties as, for example, application only to non-negative (NMF) matrices, or the timeconsuming parameter tuning (K-means). $ppmi(A)$ is factorized using SVD and decomposed into the product of three matrices:

$$ppmi(A)_{n\times m} = U_{n\times r} S_{r\times r} V^T_{m\times r}, \tag{2.3}$$

where $n \times m$ is the size of $ppmi(A)$, r is the rank of $ppmi(A)$ (in particular, sparse matrices are often of a small rank), U and V are in column orthonormal form and S is a diagonal matrix of singular values k, thus dramatically reducing the number of parameters from nm to $(n+m)f$ (Jurafsky and Martin, 2014). The matrix U is the word space consisting of tokens and the number of dimensions of SVD, and S consists of weights. In order to compare two terms v and l Deerwester et al. (1990) suggested applying the cosine of the angle between word vectors $\overrightarrow{v}$ and $\overrightarrow{l}$. The cosine metrics is based on the dot product. To prevent its favor for long vectors, the dot product is normalized by dividing it by the length of the vectors:

$$\cos(\overrightarrow{v}, \overrightarrow{l}) = \frac{\overrightarrow{v}\,\overrightarrow{l}}{|\overrightarrow{v}||\overrightarrow{l}|} = \frac{\sum_{i=1}^{N} v_i l_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} l_i^2}} \tag{2.4}$$

There are several ways to represent a word as a vector. BoW builds sparse vectors of word occurrences in texts, and as for the real tasks the length of vocabulary often runs into huge numbers, BoW model faces scalability challenges. Moreover, it does not respect semantic relatedness of words and syntax, that can potentially be the determining factor of the sentence meaning.

Latent Semantic Analysis is one of the most used methods to obtain the semantic information from co-occurrence statistics. LSA succeeds in picking up the word importance score from the data provided by the corpus and to determine the word similarity. But the computation needed to reach the goal is massive and task dependent. For example, one needs to recalculate weights whenever some new data is added into the corpus, because word frequencies are changed. In 2013 Mikolov et al. presented two algorithms in a software package word2vec for learning dense embeddings faster and easier (Mikolov et al., 2013c). Inspired by neural net language models, word2vec triggered the development of other embedding methods like Glove (Pennington et al., 2014) and Fasttext (Bojanowski et al., 2017), that are pointed in the next chapter.

Distributed Word Representations

The approaches described in the previous section were (and still are) good, when the number of texts is small and the dictionary is restricted. Although, as it is mentioned above, they have significant limitations - a great variety of texts appear every day in the Web, and these texts have an expanding vocabulary. Previously known models are not able to cope with such a volume of texts. To illustrate: the number of words in English is very roughly a million (Michel et al., 2011) - the matrix of joint occurrences of only word pairs is $10^6 \times 10^6$. Even today, such a matrix hardly fits into the memory of computers, and 10 years ago it was not possible at all. Of course, many methods were invented to simplify the processing of such matrices (eigendecomposition, Schur decomposition, SVD etc.), but all of these methods have their own drawbacks.

The idea of using distributed word embeddings for statistical language modelling was suggested in 2003 by Bengio et al.. They trained the word representations in a neural language model adding millions of the model's parameters. The results outperformed the before-used n-gram models, but suffered from very challenging computation. **Skip-gram** and **Continuous Bag of Words** (CBOW) are two approaches of the word2vec family (Mikolov et al., 2013a,c). The former predicts surrounding context words given a word, while the latter guesses the word based on its context (see Figure 2.2). Unlike its predecessors word2vec builds dense vectors based on probabilities and wins from a big amount of data. Instead of counting the word occurrences in a corpus, wor2vec trains a binary classifier, asking if a target word w_t comes up near some other word w_i in a corpus. It focuses not on accuracy of this prediction task, but on representing the collected weights as word embeddings.

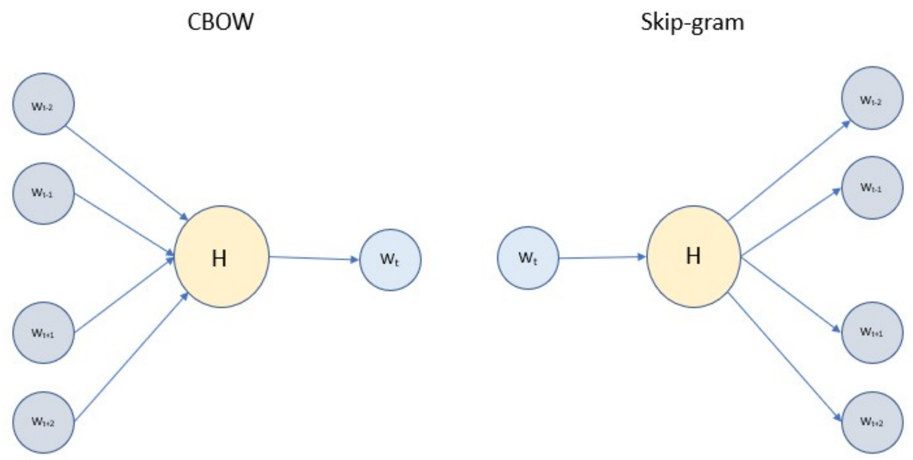


Figure 2.2: CBOW and Skip-gram models presented by Mikolov et al. (2013c).

Word2vec is a simple neural network with an input layer, a hidden projection layer and an output layer. Skip-Gram takes a numerical representation of w_t (usually a one-hot encoded vector) as an input layer, the hidden layer consists of hidden neurons, or features. There is no activation function in the projection layer. The output layer is a probability distribution of what words are likely to be seen in the w_t's context. It is often computed using the softmax function:

$$P(w_{t+j}|w_t) = \frac{\exp(\overrightarrow{w}_{t+j}, \overrightarrow{w}_t)}{\sum_{i=1}^{|V|} \exp(\overrightarrow{w}_i, \overrightarrow{w}_t)}, \tag{2.5}$$

where $\overrightarrow{w}_t$ and $\overrightarrow{w}_i$ are the target and the context word embeddings of words w_t and w_i.

Softmax changes all the weights every time a new observation (out of thousands) is made, that is why it is expensive to compute. Based on the fact that most words do not occur together, the biggest part of softmax calculations are redundant. To solve this, a simplification of Noise Contrastive Estimation (Gutmann and Hyvärinen, 2012) called **Negative Sampling** was proposed. The idea of the approach is to maximize the probability of encountering the target word among positive samples (context words), and to minimize it in negative samples (not context words) simultaneously.

$$P(w_{t+j}|w_t) = \log \sigma(\overrightarrow{w}_{t+j}, \overrightarrow{w}_t) + \sum_{i=1}^{k} E_{w_i \sim P_n} \log \sigma(-\overrightarrow{w}_{t+j}, \overrightarrow{w}_t), \tag{2.6}$$

where k is the number of negative samples, and P_n is the unigram noise distribution.

Although Skip-gram and CBOW models are often referred to as opposites, they have more in common than in difference. The input layer of CBOW consists of numerical representation of the C number of context words and the output layer is the inverse softmax function:

$$P(w_t|w_{t-C}, ..., w_{t-1}, w_{t+1}, ..., w_{t+C}) = \frac{\exp(\overrightarrow{w}_t, \overrightarrow{w}_s)}{\sum_{i=1}^{|V|} \exp(\overrightarrow{w}_i, \overrightarrow{w}_s)}, \tag{2.7}$$

where $\overrightarrow{w}_s$ is the sum of word embeddings of context words.

Stanford **Global Vectors** (GloVE) model (Pennington et al., 2014) combines methods of the two main word vector models word2vec and LSA. The first stage is the collection of statistical information by defining text frequency matrix (inspired by LSA). Like word2vec, GloVe is based on a context window that moves over the corpus collecting information about co-occurrences of the words. But in contrast to word2vec, they refer not to probability, but to global statistics of the corpus. They use matrix factorization (like LSA) to build a dense co-occurrence matrix, and apply a sum of squared minimization between the dot product of $\overrightarrow{w_t}$ and its positive samples and the log of their co-occurrences:

$$A_{GloVE} = \sum_{t,c=1}^{|V|} f(X_{tc})(\overrightarrow{w}_t, \overrightarrow{w}_c + b_t + b_c - log(X_{tc}))^2, \tag{2.8}$$

where $\overrightarrow{w}_t$ and $\overrightarrow{w}_c$ are word vectors corresponding to the target word w_t and its context word w_c, $f(X_{tc})$ is fulfilling a weighting function that prevents the model overfitting. b_t and b_c are the biases.

The smallest unit one can feed into both GloVe and word2vec are individual words. **FastText** (Bojanowski et al., 2017) is an extension to word2vec, it has the same training process but instead of feeding whole words into neural network, it uses n-grams. It is useful when working with morphologically rich languages such as Finnish or Ukrainian. Such languages have many inflected forms which occur rarely in the training corpus and are hard to learn properly. Moreover, it is beneficial, when working with corpora containing misspellings, like Twitter. For example, the word embedding for the word *laptop* will be represented as a sum of its n-grams, for instance, tri-grams - *lap, apt, pto, top*:

$$\overrightarrow{w}_t = \sum_{g \in G_{w_t}} \overrightarrow{g}, \tag{2.9}$$

where G_{w_t} is a set of n-grams appearing in the w_t and $\overrightarrow{g}$ is a vector representation of each n-gram g. Thus, rare and misspelled words can be properly represented, as it is likely that there are other words that share some of their n-grams.

2.1.3 Crosslingual Word Embeddings

The collection and annotation of data are cost- and labour-intensive and time-consuming, that is why data available for under-resourced languages (according to Klementiev et al. (2012), all except of English) is very limited. Inspired by the assumption that the semantic structure of one language can be obtained from another one, various approaches have been proposed to learn crosslingual embedding mappings by either joint training crosslingual word representations using parallel data (Chandar et al., 2014; Hermann and Blunsom, 2013) or by mapping monolingual word embeddings to a common space using linear transformation (e.g. Mikolov et al. (2013b); Lu et al. (2015); Barone (2016); Artetxe et al. (2017); Conneau et al. (2017)).

Both Hermann and Blunsom (2013) and Chandar et al. (2014) used aligned sentences by encoding them as a sum of their word embeddings:

$$\vec{s} = \sum_{i=1}^{n} w^i \tag{2.10}$$

The former aims at minimizing the distance between $\vec{s}_t$ and $\vec{s}_t$ (source and target sentences respectively) by assigning similar vectors to aligned sentences and then optimize this distance by maximizing it for negative samples $\vec{s}_{neg}$:

$$A = \sum_{\vec{s}_s, \vec{s}_t \subset \vec{C}} \sum_{i=1}^{n} egmax(0, 1 + E_{dist}(\vec{s}_s, \vec{s}_t)) - E_{dist}(\vec{s}_s, \vec{s}^{\,i}_{neg}), \tag{2.11}$$

where C is the corpus of aligned sentence representations and neg is the number of negative samples.

Instead of distance minimization Chandar et al. (2014) proposed to train a Neural network to learn a mapping from bag-of-words representations of a sentence $\vec{s}_s$ to its translation $\vec{s}_t$ (see Figure 2.3).

The main limitation of these methods is that word embeddings can be obtained just from parallel sentence-aligned data. This data is restricted, so its collection will require extra time and costs. Thus, most crosslingual mapping

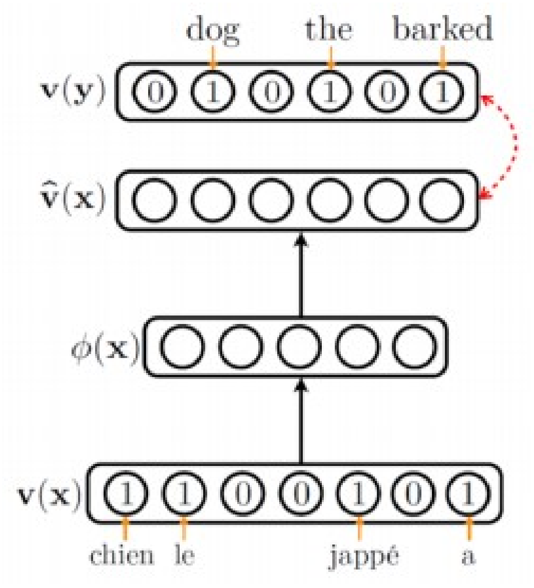


Figure 2.3: A bilingual decoder with a reconstruction error. (Chandar et al., 2014)

approaches are based on the separate training of monolingual embeddings in both languages and their linear transformation to a shared space. Most of these methods are supervised and require parallel data of different alignment and a bilingual dictionary for learning crosslingual representations. Mikolov et al. (2013b) first noticed that geometric relations between words are similar across the languages. For example, Figure 2.4 shows that representations of numbers, animals and names in German and French have similar geometric correlations inside one language. This leads to the idea of transforming the vector space of a source language $\overrightarrow{L}_s$ to the vector space of a target language $\overrightarrow{L}_t$ using a least-squares objective. They used the vocabulary of n most frequent words in L_s and their translation into L_t as seed words, and then learn the transformation matrix T by minimizing mean squared error:

$$min_T \sum_{i=1}^{n} = |T\overrightarrow{w}_s^i - \overrightarrow{w}_t^i|^2, \tag{2.12}$$

where $\overrightarrow{w}_s^i$ and $\overrightarrow{w}_t^i$ are word representations from source and target vector spaces respectively.

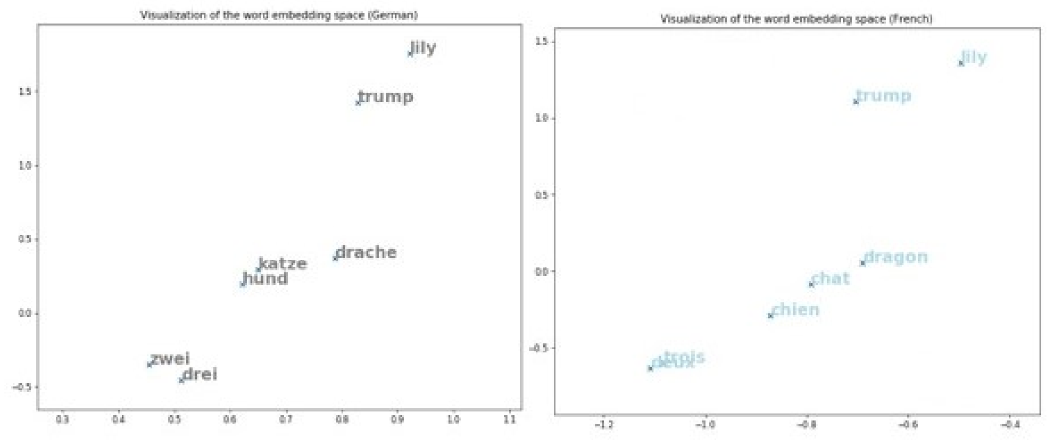


Figure 2.4: Similar geometric relations between numbers, animals and names in German and French (inspired by Mikolov et al. (2013b))

However, Dinu et al. (2014) prove that vector spaces contain *hubs* - word vectors that are nearest neighbours of many other vectors. That makes the accuracy of the method too low for the practical use. In Lazaridou et al. (2015) they suggested to estimate the mapping function with a max-margin ranking loss instead of mean squared error, which leads to the significant hubness reduction and consequently to the performance improving.

To overcome the pulling of unnecessary information that does not correlate across the languages from monolingual data sets and the increasing dimensionality of aligned vectors, Faruqui and Dyer (2014) used another approach for learning crosslingual mapping - canonical correlation analysis (CCA). In contrast to Mikolov et al.'s method, CCA learns the transformation matrix for each language separately. Firstly, two new matrices are defined: $\overrightarrow{L}'_s \subset \overrightarrow{L}_s$ and $\overrightarrow{L}'_t \subset \overrightarrow{L}_t$, where every word in $\overrightarrow{L}'_s$ has its direct translation in $\overrightarrow{L}'_t$. Given seed vectors $\overrightarrow{w}^i_s$ from $\overrightarrow{L}'_s$ and $\overrightarrow{w}^i_t$ from $\overrightarrow{L}'_t$ two projected vectors are then obtained:

$$\begin{aligned} \overrightarrow{w}^{i(P)}_s &= \overrightarrow{w}^i_s y_i \\ \overrightarrow{w}^{i(P)}_t &= \overrightarrow{w}^i_t x_i, \end{aligned} \tag{2.13}$$

where y_i and x_i are two projection directions. The correlation of the projected vectors can be then defined as:

$$p(\overrightarrow{w}_s^{i(P)}, \overrightarrow{w}_t^{i(P)}) = \frac{cov(\overrightarrow{w}_s^{i(P)}, \overrightarrow{w}_t^{i(P)})}{\sqrt{cov(\overrightarrow{w}_s^{i(P)2})(cov(\overrightarrow{w}_t^{i(P)2}))}} \tag{2.14}$$

CCA aims to maximize p and, as a result, outputs the projection vectors y_i and x_i:

$$y_i, x_i = CCA(\overrightarrow{w}_s^i, \overrightarrow{w}_t^i) = \underset{y_i, x_i}{\operatorname{argmax}} \, p(\overrightarrow{w}_s^i y_i, \overrightarrow{w}_t^i x_i) \tag{2.15}$$

Equation 2.15 and Equation 2.13 prove, that the entire vocabulary of source and target languages can be projected, therefore solving both the dimensionality problem (as transformation vectors cannot appear longer than original vectors) and the out-of-vocabulary words problem (as CCA is applied only to the words that have their direct translation in the dictionary).

In 2015 Xing et al. noted that there is inconsistency in measuring the relatedness of words in the work of Mikolov et al. (2013b), as the latter used dot product as the comparison function for learning word embeddings and cosine similarity as the evaluation function. And they differ significantly. To cope with the disparity Xing et al. suggested to redefine the optimization function of Mikolov et al. (2013b) (see Equation 2.12) by enabling length normalization of word embeddings during the training. In particular, this entails, that the inner product coincides with the cosine distance:

$$max_T \sum_{i=1}^{n} (Tw_s^i)^T w_t^i \tag{2.16}$$

This results, though, in another problem. To preserve the unit length after mapping, the projected vector Tw_s^i has to be normalized. It is satisfied by constraining the transformation matrix T as orthogonal ($TT^T = I$, where $T^T = T^{-1}$). The same orthogonality approach is used in the work of Artetxe et al. (2016), where the importance of orthogonality was empirically proved, and in the work of Zhang et al. (2016), where crosslingual word embeddings for POS tagging were learned.

Since then a related research line has been aimed at improving the mapping of crosslingual word embeddings (Shigeto et al., 2015; Smith et al., 2017), but they all have expensive resource requirements such as bilingual parallel data, aligned at word, sentence or document level or a bilingual dictionary. This is unfortunate for low-resource languages. A fully unsupervised approach was proposed by Zhang et al. (2017). On the assumption that only distributional relations between languages are strong enough to learn qualitative word embeddings, they applied an adversarial autoencoders using two sets of monolingual word embeddings trained on thematically similar corpora only. They followed the standard training procedure of deep adversarial models introduced in Goodfellow et al. (2014) illustrated in Figure 2.5:

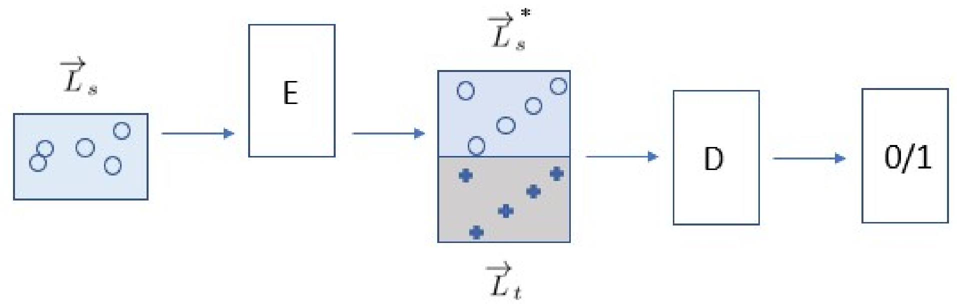


Figure 2.5: Illustration of the adversarial approach adapted from Zhang et al. (2017)

The discriminator D is trained to distinguish between the transformed embeddings $\vec{L}_s^*$ and the target embeddings $\vec{L}_t$ and to produce the binary prediction about the language of the input sentence (0 corresponds to the source language and 1 corresponds to the target language), while the encoder E is trained to prevent D from making accurate predictions, making $\vec{L}_s^*$ and $\vec{L}_t$ as similar as possible. D is a feed-forward Neural network with one hidden layer and cross-entropy loss function (Equation 2.17). The encoder loss is described in Equation 2.18.

$$L_D = -log(D(\overrightarrow{w}_t) - log(1 - D(E\overrightarrow{w}_s) \tag{2.17}$$

$$L_E = -log(D(E\overrightarrow{w}_s)) \tag{2.18}$$

However, the idea of adversarial networks for crosslingual mappings sounded encouraging, Zhang et al. (2017) reported the results to be not competitive with supervised methods. The further experimentation for model improvement was required. In 2017 Conneau et al. introduced an unsupervised domain-adversarial approach that was on par, or in some cases outperformed the existing supervised techniques. Same as Zhang et al. (2017) they followed the adversarial approach of Goodfellow et al. (2014) (see Figure 2.5), but instead of aligning all the words in the corpus, they applied the Procrustes algorithm only to the most frequent words. This method aims to minimize the shape distance (illustrated in red in Figure 2.6) between aligned source and target embeddings. They noted that the quality of embeddings for rare words is not as good as for frequent ones, which makes them harder to align. To refine the mapping, they proposed to build the dictionary of mutual nearest neighbors of the most frequent words.

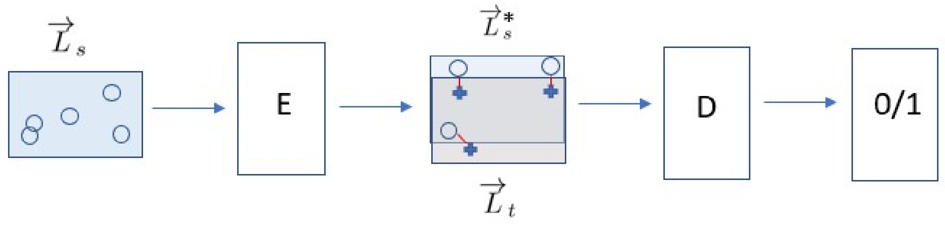


Figure 2.6: Illustration of the adversarial learning with the Procrustes algorithm adapted from Conneau et al. (2017)

In their work Conneau et al. succeeded to prove the possibility to obtain the crosslingual word embeddings of high quality by using monolingual corpora only. The effectiveness of their method was demonstrated on several evaluation tasks such as semantic word similarity, sentence translation retrieval and word translation for different language pairs, where one language of the pair was always English. The approach showed no suffering of performance degradation compared to the supervised methods. In this work we adapt the model (Conneau et al., 2017) for a German-French pair. We go from the word and sentence to the document level, exploring the performance of crosslingual word embeddings on tasks facing deeper semantic challenges - text categorization, sentiment analysis and humour detection.

2.2 Text Classification as Evaluation of Crosslingual Word Representation Models

2.2.1 Introduction to Text Classification

Nowadays a huge amount of data has been accumulated by various repositories of knowledge materials (such as offline libraries and the Internet). The problem lies in the difficulty of orientation in this range of textual data. The inability to obtain the most relevant and complete information on a specific topic makes most of accumulated resources useless. Since the study of a specific task requires more and more work for the direct search and analysis of information on the topic, many decisions are based on an incomplete view of the problem.

Automatic text classification is one of the fundamental tasks of NLP (Sebastiani, 2002). It aims to assign labels, tags or categories to the textual data according to its insights. However, the unstructured nature of textual data makes it a challenging and time-consuming task.

Formally, text classification can be described as approximation of some unknown efficiency function $F : D \times C \to \{0, 1\}$ by the classifier $\overline{F} : D \times C \to \{0, 1\}$, where $C = \{c_1, c_2, ..., c_n\}$ is a set of categories, and $D = \{d_1, d_2, ..., d_n\}$ is a set of documents (Sebastiani, 2005).

$$F(d_j, c_i) = \begin{cases} 0, \text{ if } d_j \not\subset c_i \\ 1, \text{ if } d_j \subset c_i \end{cases} \tag{2.19}$$

There are two assumptions, that are usually considered when solving the problem of classification (Sebastiani, 2002):

- Categories are only character labels with no meaning.
- No additional sources of data, except the text of the document are used for classification. In particular, there are no files with metadata of documents (publication date, type of document, etc.).

Binary classification is illustrated in Example 2.19. However, solving it means also solving the multi-class classification by defining one binary classifier for each $c \subset C$. In the case, although, the categories must be independent.

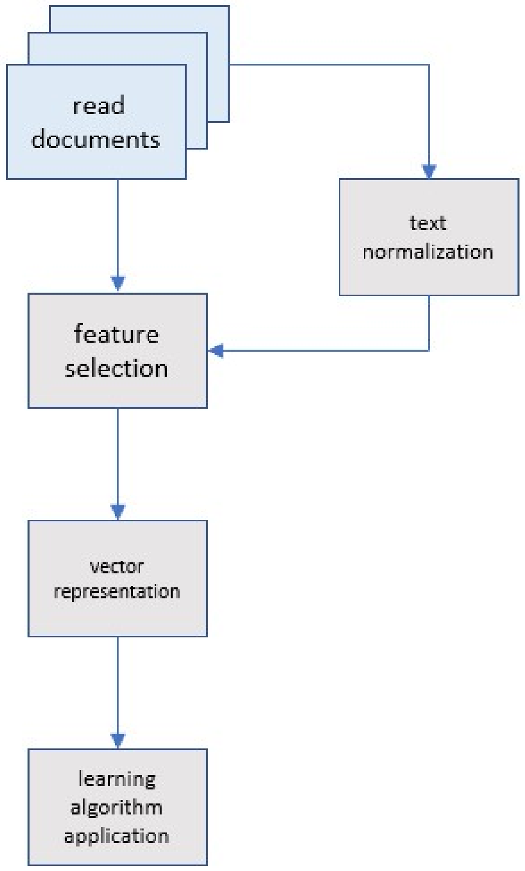


Figure 2.7: Illustration of the document classification process

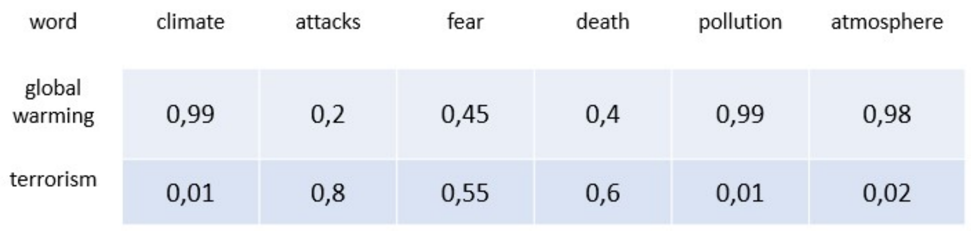
Text classification has broad applications such as topic labelling, sentiment analysis, machine translation (resolving homonymy), language detection, spam detection, etc. There are a lot of successful approaches presented in literature for any of these tasks (for example Turney (2002); Zhang et al. (2012); Pang et al. (2002); Hofmann (2001); Gao et al. (2010)). They can be grouped into:

- rule-based systems
- machine learning systems:
  - classifiers
  - neural networks
- hybrid systems

The first rule-based attempt was introduced by Maron in 1961. It was based on a set of handcrafted if-then-else rules. Since then a number of applications have been made to improve the performance, and if in the beginning it was mostly based on the frequency count (meaning, for example, that if the text has more words related to art than to history, it is classified as art-related), later such systems became more complex. For instance, the RIPPER rule learning algorithm (Cohen, 1995) succeeded (in principle) to formulate a classifier that depends on word order, and Sasaki and Kita (1998) enriched it by using the hierarchical categories. The best results were obtained by Hayes et al. (1990). Their set of rules reached a 90% accuracy in text classification and outperformed even the machine learning approaches of the late 90s.

This approach can be a good option if working with a small collection of documents that can be easily controlled and carefully analyzed by human. But it has obvious disadvantages. In order to select words that are significant for classification, it is necessary to have deep knowledge of the domain. Moreover, the presence or absence of a word is not always the decisive factor for making a decision. The rules can be complicated by adding nested if-clauses. But, firstly, a person's ability to formulate such rules is very limited, because the complexity of the rules grows exponentially with the number of words selected for classification, and, secondly, the approach requires the high cost of human power for defining the rules.

In the 1990s a more reasonable method for text classification was suggested. The idea was to choose a certain weight for each word, meaning what category this word is more likely to correspond. Machine learning (ML) is a supervised method of classification, where the learner (automatic builder) builds a classifier for the category c_i, based on the observation obtained from the manually labeled training data set. It aims to collect and note the characteristics a new test document should have in order to belong to c_i.



word	climate	attacks	fear	death	pollution	atmosphere
global warming	0,99	0,2	0,45	0,4	0,99	0,98
terrorism	0,01	0,8	0,55	0,6	0,01	0,02

Figure 2.8: A weight-term classification matrix

Mitchell (1996) provided a comprehensive introduction to all machine learning algorithms, noting that building a text classifier does not differ a lot from other tasks of ML. The main difference is the representation of the document (Leopold and Kindermann, 2002). The number of features (usually words) is often very high, that can cause problems in qualitative learning. Since the detailed overview of the modelling of ML classifiers is beyond the scope of this work, we suggest the paper of Sebastiani (2002). He undertook the analysis of all aspects of machine learning classifiers for text classification tasks, such as approaches for dimensionality reduction, feature extraction, construction and evaluation of ML text classifiers.

Depending on the domain, classification task and training data, different algorithms reach better performance. Kim et al. (2002) argued Naive Bayesian generative classifier family to be the most effective, albeit simple, methods for text tasks. Naive Bayes classifier is based on Bayes Theorem and assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It computes the conditional probabilities of two events together, given probabilities of each event individually.

$$P(c_i|d_j) = \frac{P(d_j|c_i)P(c_i)}{P(d_j)}, \tag{2.20}$$

where $P(c_i|d_j)$ is the probability that document d_j belongs to category c_i, $P(d_j|c_i)$ is the probability to meet d_j among all documents of c_i, $P(d_j)$ is the probability of d_j being in a corpus, and $P(c_i)$ is the probability to meet a document of c_i in a corpus. Robertson and Harding (1984) were arguably the first ones to use the model for document indexing. Lewis (1992) investigated the effectiveness of the approach and proved that the performance of the method reaches its peak at a higher feature set size. Kłopotek and Woch (2003) presented results of empirical evaluation of a Bayesian multinet classifier based on a new method of learning very large tree-like Bayesian networks. Forman and Cohen (2004) compared the performance of different classifiers when having little training data. Naive Bayes outperformed other models trained in the same circumstances.

The Logistic Regression classifier, also known as a Maximum Entropy classifier, calculates the conditional probability distribution of a class based on the given data set. It assumes no previous knowledge, thus expecting that the distribution is uniform which means that it has the Maximum Entropy. The experiments of Nigam et al. (1999) depicted that logistic regression sometimes performs better than Naive Bayes. The comparative analysis of generative and discriminative models (Ng and Jordan, 2002) described the same results in a more profound way. They showed that logistic regression (discriminative model) outperforms generative Naive Bayes, when the training size is increased. They mathematically and empirically proved error properties of both models, noting, that although the generative model reaches its asymptotic error faster ($O(log(n))$) than the discriminative model ($O(n)$), the generative model reaches the asymptotic solution for fewer training sets than the discriminative one.

Alternatively to these, there are a lot of other machine learning algorithms for classification tasks discussed in academia (for example, k-nearest neighbor (Yang, 1999), support vector machines (Joachims, 1998), decision tress (Lewis and Ringuette, 1994), etc.). Different space and time considerations, data, output, classification tasks influence the performance of classifiers, and despite the decades of research, none of them has proved to always outperform others.

Neural Networks

The human brain has a big advantage over the computer. A person can recognize faces, even if there are many extraneous objects and poor lighting in the room, or easily understand strangers in a noisy room. But the most amazing feature of the human brain is that it can learn. No software and no updates are needed if a person wants to learn to ride a bike. Machine learning algorithms follow the function learned from the data, but at some point, they still need human guidance. For example, if a ML algorithm gives an inaccurate outcome, human is the one to make system adjustments. Recently, new systems that are capable enough to learn and adapt themselves were proposed to use for text classification.

In 1943, McClulloch and Pitts modeled an artificial neuron that received information from other neurons, and, depending on the common input weight, either was activated or remained inactive. In the 1960s, it was proved that such neural models have properties similar to the brain: they can perform complex pattern recognition operations, and they can function even if some connections between neurons are broken. The demonstration of the perceptrons of Rosenblatt (Rosenblatt, 1961) showed that simple networks of such neurons can be trained on examples. Later, Minsky and Papert (1969) proved that simple perceptrons can solve only a very narrow class of linearly separable problems. This led to degradation of activity in neural networks research till the 1980s, when a number of articles by Rumelhart et al. (Rumelhart et al., 1985, 1988) were presented. In these works Rumelhart et al. specifically addressed the issues and suggested solutions for the problems described in Minsky and Papert (1969). It helped to rerise research interest for artificial neural networks and already in 1989, the multi-layer perceptron (MLP), composed of an input, output and one or more hidden layers, could implement any even nonlinear function (Hornik et al., 1989).

The method consists of two passes: forward and backward. In the forward pass, a training signal is sent through all layers from input to output. The sum of the input weights produces the activation signal that is passed to the activation function to obtain one output from the neuron. By subtracting the decision of the output layer from the ground truth labels, an error signal is determined. In the backward pass, the error signal propagates in the opposite direction, from output to input. At the same time, synaptic weights are adjusted in order to minimize this error. A detailed description of the method can be found in a variety of sources (for example Haykin (1994)) .

In the last decade, Artificial Neural Networks (ANN) have been applied to nearly any task related to analysis of natural information, such as language, speech, image, video, etc. They have also demonstrated the great performance for many NLP tasks (Goldberg, 2017; Liu and Zhang, 2018). Convo-

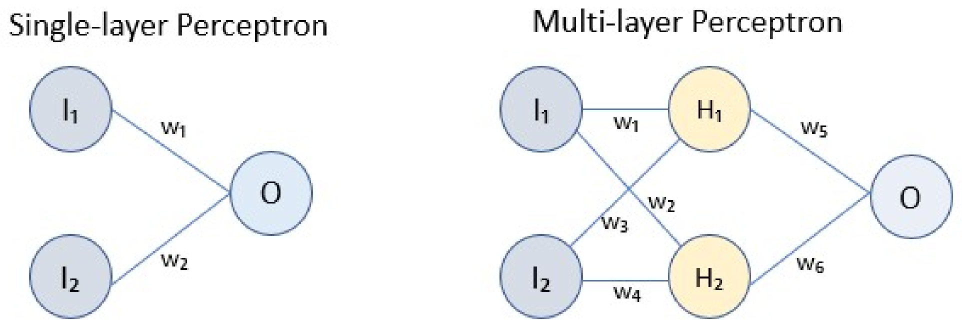


Figure 2.9: Schematic illustrations of single-layer and multi-layer perceptrons, where I$_n$ are the input neurons, O is the output neuron, H$_n$ are the hidden neurons and w$_n$ are the weights

.

lutional Neural Network (CNN) and Recurrent Neural Network (RNN) are the most widely used types of ANN architectures, particularly for text classification tasks. As illustrated in Figure 2.10, RNN hardly differs from MLP (see Figure 2.9), except for some kind of cyclic connection. The hidden layer sends its own weights back to itself, which allows the network to remember the previous input. RNNs are attractive, because they are potentially able to associate previous information with the current task, for example, knowing the part of speech of the previous word can help in defining the part of speech of the current word. But there are cases, when more context is needed, for instance, the task is to predict the last word of the sequence *"Georgia is my second motherland. I can fluently speak ?"*. The closest context suggests that the last word will be the name of the language, but in order to establish which language in particular, the context of *Georgia* from a more distant past is required.

Thus, the gap between the actual information and the point where it is needed can become very large. Unfortunately, as this distance grows, RNNs lose their ability to bind information. This problem was thoroughly studied by Hochreiter (1991) and Bengio et al. (2003). In 1997, Hochreiter and Schmidhuber presented Long short-term memory (LSTM) architecture (see Figure 2.11). LTSM is a special kind of RNN architecture, designed to memorize information for long periods of time. Moreover, LTSMs can detect important information and forget irrelevant data. It becomes possible because the visualization of a usual neuron is replaced by some kind of network (see

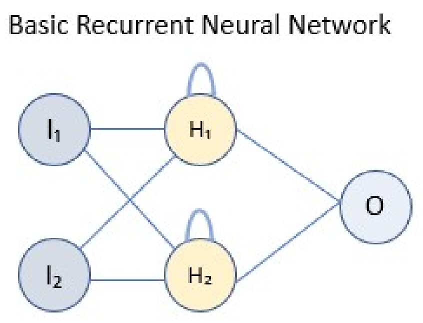


Figure 2.10: Recurrent Neural Network

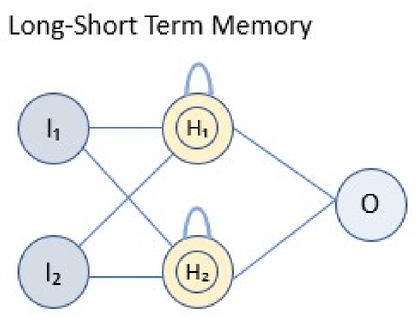


Figure 2.11: Long Short-Term Memory

Figure 2.12), that has memory cells and three gates: input gate, forget gate and output gate, which control when this memory needs to be reset, rewritten or saved, etc. These gates are also trained in the same way as everything else. In their work, Sak et al. (2014) applied deep LSTM architectures to large

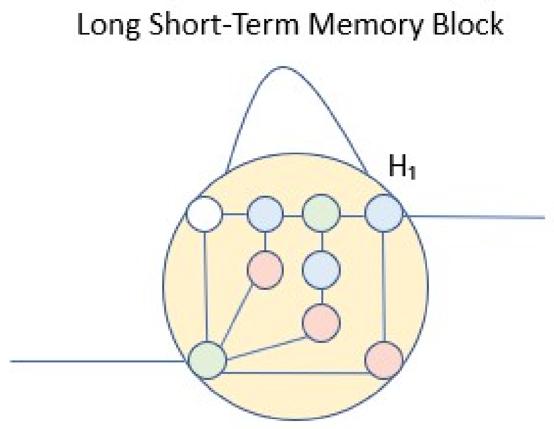


Figure 2.12: Architecture of the hidden layer neuron (also known as LSTM block) inspired by Otter et al. (2018)

scale acoustic modeling, achieving state-of-the-art performance. Later LSTM was trained on part-of-speech (POS) tagging, reaching a 97,40% tagging accuracy (Wang et al., 2015). Liu et al. (2016) used the multitask learning framework to jointly learn across three related text classification tasks. Dou

(2017) used LSTM to learn a document representation as the first part of the rating prediction task.

Convolutional neural networks were not originally considered to work with textual data, they were used in computer vision and pattern recognition. The architecture of CNNs is different from other ANNs: the layers are organized in three dimensions: width, height and channels (see Figure 2.13) and the neurons in one layer are not connected with all the neurons in the next layer but only with a small region of the layer. Each layer applies different filters and combines their result. Another difference in the building of CNNs is pooling layers. They are usually applied after the convolutional layers and used to reduce the output dimensionality while keeping the most relevant information.

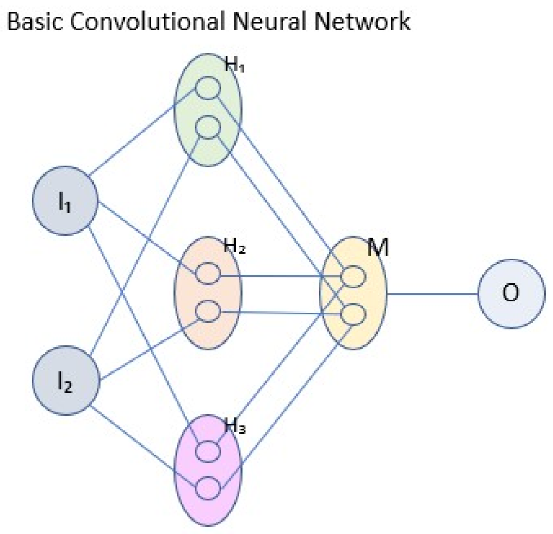


Figure 2.13: Basic architecture of Convolutional Neural Network, where H$_n$ are convolutional hidden neurons and M is a maxpooling hidden neuron.

Collobert et al. (2011) first suggested applying CNNs for language processing tasks, but since the representation of text is an embedding matrix, it has only one dimension - width. Therefore, in order to distinguish it from standard convolution, they proposed one-dimensional (1D) convolution. Each row of the matrix corresponds to one token (usually word). During the training, a number of convolutional filters of different widths slide over the entire word embedding matrix row by row. As the size of text documents always varies, the output of convolutional layers is also variable-sized. To produce the fixed-sized output required by a fully connected output layer (O in Figure 2.13), like in computer vision, a pooling layer is used.

Kim (2014) evaluated a CNN model for Sentiment Analysis and Text Categorization tasks. However, the architecture they applied is quite simple, the model achieves good performance overall, and outperforms all the existing models on some of the data sets. The input layer is a sentence comprised of concatenated word2vec word embeddings. It is followed by a convolutional layer with multiple filters, a max-pooling layer, and a softmax classifier. Johnson and Zhang (2014) studied CNN on text categorization task, applying it on high dimensional hot-vector matrix (instead of dense vectors such as word2vec or GloVE). They also proposed a space-efficient bag-of-words-like representation for the input data, reducing the number of parameters the network needs to learn. The experiments of Conneau et al. (2016) showed that networks with a large amount of convolutional neural networks work better for text classification tasks. The comparative analysis of LSTMs in CNNs in NLP illustrated, that support CNN outperforms LSTM for classification of textual data (Adel and Schütze, 2016; Yih et al., 2016). They admitted that RNNs in general and LSTMs in particular have presented the best results when working with sequences of words and paragraphs, such as sequence prediction.

Collobert et al. (2011) introduced deep neural networks for traditional NLP tasks, and hereby sparked the deep learning discussion in NLP academia. Since then the application of ANNs for text tasks has been increasing. However, the lack of high-quality labeled training data in non-English languages, causes either the inability to apply those methods to other languages, or the need to experiment with limited data sets, that leads to superficial analysis of the approaches (for example Lee and Renganathan (2011)). This fact has become the motivation for crosslingual text classification, inferring to obtain the knowledge from training data in a resourced language and to apply it for testing in an under-resourced language.

In the following sections we briefly describe the existing research that has been done towards text categorization, sentiment analysis and humour detection within one language and crosslingually.

2.2.2 Text Categorization

The task of text categorization[2] aims to assign a document to one or several topics based on its context. This task has important applications in real life. For example, news articles are usually divided according to topics, and e-mail messages are divided into “spam” and “not spam”.

Klementiev et al. (2012) induced crosslingual representations of words in English and German languages separately and aligned them into a shared space, making use of parallel data. To evaluate the performance, they trained a simple perceptron algorithm on 10 000 news articles (the subset of the Reuters corpora (Lewis et al., 2004)), where each document is assigned to a single category. The classifier reached a 77.6% accuracy with English data set as a training set and German data used as a test set, and 71.1% when vise verse. All the existing research in the sphere of crosslingual document classification is directed at comparing new ways to align word embeddings, that is why the same data set and experimental setup as in Klementiev et al. (2012) were used for evaluation. The state-of-the-art results were obtained by Chandar et al. (2014) (see the description of the model in Section 2.1.3). Their approach reached a 91.8% accuracy for an English-German pair.

Mogadala and Rettinger (2016) and Luong et al. (2015) expanded the research and trained the classifier also for English-Spanish and English-French pairs. The transfer, however, achieved the disappointing ca 80% for English-French and 60% for English-Spanish.

In their newly published work, Artetxe and Schwenk (2018) proposed another evaluation method with no transfer scenario. They trained a single encoder with a shared vocabulary for 93 languages. To evaluate, a feed forward multi-layer perceptron with one hidden layer to classify and optimize monolingual data (in the work - English) is trained. The results then can be evaluated on any other language with no modification. They demonstrated the approach on the data from a new Reuters subtract presented in Schwenk and Li (2018). It consists of news articles in 8 languages, including syntactically and morphologically different from English Russian, Japanese and Chinese. The approach gains a competitive performance, obtaining, for instance, an 84.78% accuracy for testing in German, 77.33% for Spanish and 71.93% for Chinese.

[2]In the work we consider terms *text/document categorization*, *topic spotting* and *text/document tagging* interchangeable.

2.2.3 Sentiment Analysis

The task of sentiment analysis, also known as opinion mining, is, on the one hand, technically very difficult, but, on the other hand, useful. Businesses are always interested in finding public opinions about their products, it helps them improve the existing product or understand how they can increase the target audience. Customers search for previous users' experience, before purchasing a service or a product. Opinion mining finds its practical application in various research fields: sociologists collect data from the social networks (for example, religious beliefs), political scientists use information about political views of the population from blogs, marketers analyze Twitter to find out which notebook model succeeds in the marketplace, psychologists define depression of social networks' users. To introduce the challenge of the task we use the following review:

"a) I hate to admit it, but these movies are mesmerizing. b) The acting is just so-so, the story is a little convoluted and to top it all off the friggin vampires sparkle in the sunlight. c) All that being said, you really can't help being drawn into this world and these characters. d) The movies may lack a little when it comes to vampire lore, but the whole package somehow just works. e) The high-def transfers look really good, and my only complaint about the discs is that the dialogue always seems overpowered by the score, even when run through a home theater system." [3]

First of all, the review does not have explicit opinion about the movies. Even breaking it into sentences, does not always help. Sentences *b)* and *d)* express some negativity, while sentences *a), c)* and *e)* represent positive or mixed emotions. Secondly, however, the target of the review is to share the opinion about the movie in general, it consists of several subopinions: sentences *a), c)* and *d)* describe the thoughts on the movie as whole, sentences *b)* and partly *d)* review actor playing and the script, sentence *e)* speaks about technical qualities of DVDs.

The sentiment classification within one language was studied extensively in literature in the last decade, the survey of the most successful approaches is provided by Pang et al. (2008). However, the idea of making use of high-quality data in one language and applying it to data in another one came into focus just a few years ago. Zhou et al. (2016) evaluated and compared

[3]An amazon review posted by S. Carlson on March 3, 2012

different existing approaches (Prettenhofer and Stein, 2010; Tang et al., 2012; Xiao and Guo, 2013; Pham et al., 2015) for crosslingual matching on crosslingual sentiment analysis tasks (all of these approaches require parallel data for training word representations). They used the multilingual multi-domain Amazon review data set created by Prettenhofer and Stein in 2010. In addition, they proposed their own cross-lingual representation learning model BiDRL which learns both the word and document representations in a target and a source language simultaneously. For this, however, they needed the translation of the entire training and test sets. In contrast to previous works, which used the semantic relatedness between parallel sets only, the algorithm employed sentiment and semantic correlations. This helped to outperform all the approaches, reaching the state-of-the-art accuracy of ∼84% testing in German, ∼83% in French and ∼76% in Japanese.

2.2.4 Humour Detection

Many have tried to comprehend the specifics of humour. Raskin was one of them. In his work (Raskin, 1985), he sheded light on semantic aspects of humorous phenomena, concluding that objective laws of humour are directly connected to the properties of language and interpersonal communication. It seems obvious that a rare joke exists in isolation from thinking or communication, but Raskin uncovered deeper layers of understanding humour. In fact, he was the first to define the humour approach as linguistic, by arguing that linguistic origins are clearly revealed in humour. Prior to this, psychoanalytic developments and attitudes prevailed. Raskin did not seek to embrace humour entirely, but focused on its verbal component.

Being a necessary part of verbal communication, humour stays one of the most puzzling research fields, especially for NLP academics. Recently computers have changed their roles in human's life: from basic doers of assigned tasks they grew into intelligent agents that can interact with people, learn, read, talk, make conclusions, understand. Handling humour would raise human-machine communication to a new level of performance. But the task of humour detection (also humour recognition) is very challenging. First of all, humour is not universal, it is always based on some personal, situational or even territorial factors: a racist remark in the South of the US can still provoke laughter; a related to the situation quote from the movie is funny only to those who watched the movie.

The first attempts to build an automatic humour recognizer were made by Mihalcea and Strapparava in 2006. Their system was trained to identify funny one liners such as *"Beauty is in the eye of the beer holder"* from proverbs and news headlines. They experimented with different stylistic features suggested by Ruch (2002) as "potentially good indicators of humour" - alliteration, antonymy, and adult slang. Surprisingly, the best results were achieved by using a count based analysis only, when none of these features was added.

Yang et al. (2015), inspired by the work of Mihalcea and Strapparava (2006), took a step forward and tried to determine semantic structures behind humour, namely incongruity, ambiguity, interpersonal effect and phonetic style. Moreover, they believed in existence of laughter boosters - words that potentially make a text funny. In their work they called such words humour anchors. As the example, they provided a following sentence: *"I am happy I know sign language; it is pretty handy."* In this case the humour anchors are the phrase "sign language" and the word "handy". While separately they are not considered humorous, their combination prompts a person, when hearing "handy", to think about hands. In addition, the set of humour anchors is inseparable - if either "sign language" or "handy" is removed, the sentence loses its humour. Yang et al. treated humour recognition as a traditional classification problem and applied a Random Forest binary classifier to distinguish between funny and not funny sentences. They reached the best accuracy (85%) by using the method that takes into account both latent structures and semantic word meanings (word2vec).

A new approach for detecting humour was proposed by de Oliveira and Rodrigo (2017). They compared the "shallow" deep learning methods like Random Forest and Naive Bayes with RNNs and CNNs for humour recognition in Yelp reviews. And although the performance of RNNs turned out to be disappointing - it did not outperform the traditional ML approaches, CNN showed the best results. It proved that there are more expressive relationships between semantics and humour, and these relations can be picked up by deep learning. Chen and Lee (2017) also found convolutional NN to be the best solution for predicting audience laughter TED talks transcripts.

Despite good results of the above mentioned works, the research of semantic humour detection is still limited, because all traditional data sets (Pun of the Day[4], One Liner Set (Mihalcea and Strapparava, 2006), 111 Knock

[4]https://www.punoftheday.com/

Knock Jokes corpora[5]) are English corpora. Moreover, the evaluations are mostly focused on a specific humour type and isolated from other research.

As we are aware, there have been no works towards crosslingual humour detection.

[5]http://www.ahajokes.com/kkn111.html

Chapter 3

System Overview

3.1 System Designing

As detailed in Section 1.2, the basic idea of this thesis is to explore the performance of crosslingual word embeddings on different application tasks. For this purpose, we investigate if word embeddings are able to obtain semantic relations between lexical features and to transfer these relations across languages; we experiment with three classification tasks; we use two opposite data sets - a structured balanced one and an imbalanced one; we study the influence of crosslingual mapping on the performance of a text classifier.

Figure 3.1 shows the workflow process of the thesis. First of all, two sets of pre-trained monolingual word embeddings are fed into a system to learn mapping. Then the text train and test files are prepossessed, tokenized and converted into numerical vectors. Meanwhile, both aligned source and target embeddings build a shared embedding layer. The training data are then transferred to a NN classifier to solve the classification problem. The test set is then used to evaluate the performance of the classifier.

The investigation of this thesis can be divided into two leaps. First, we train crosslingual word embeddings and, as a second contribution, we apply these embeddings for three classification tasks.

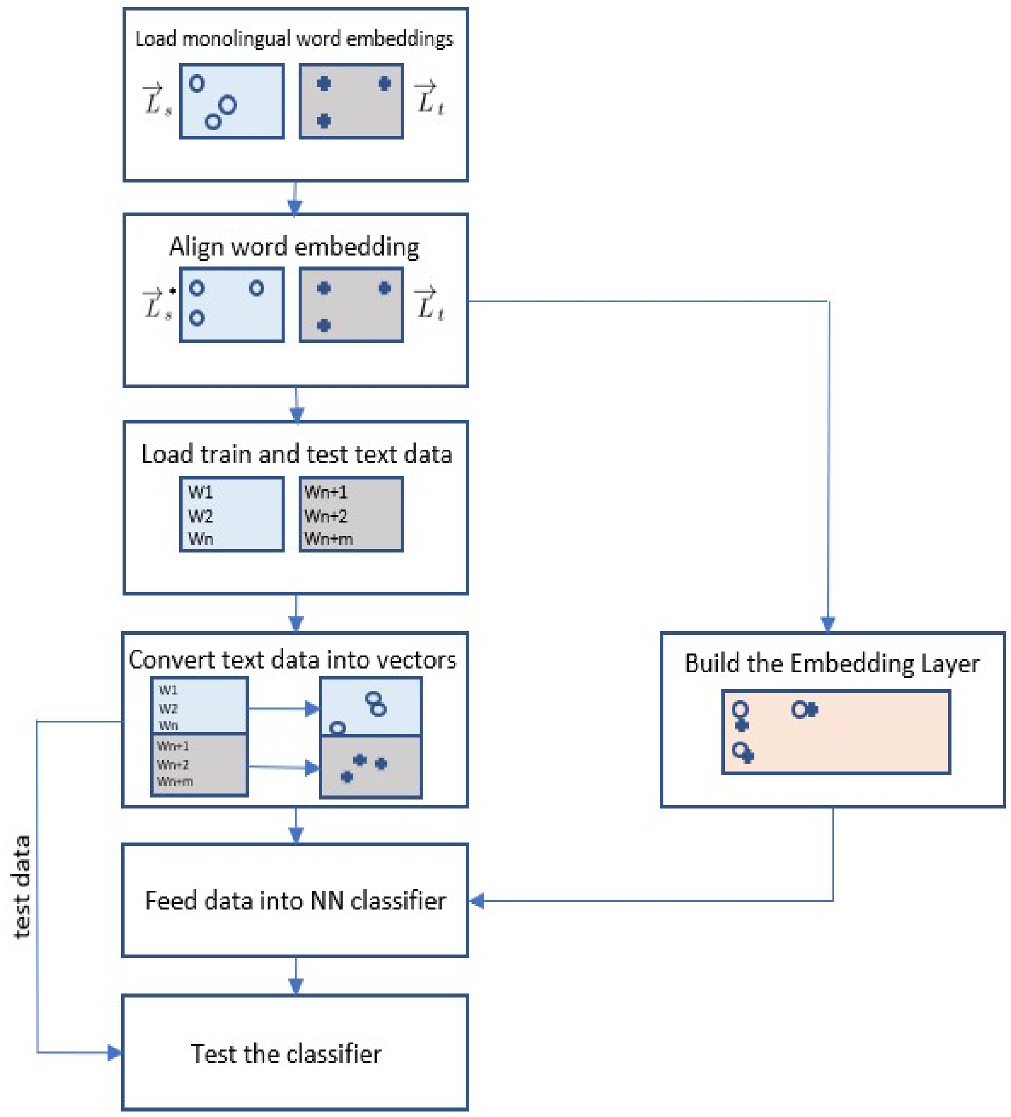

Load monolingual word embeddings
$\vec{L}_s$
$\vec{L}_t$
Align word embedding
$\vec{L}_s^{\bullet}$
$\vec{L}_t$
Load train and test text data
W1
W2
Wn
Wn+1
Wn+2
Wn+m
Convert text data into vectors
W1
W2
Wn
Wn+1
Wn+2
Wn+m
Build the Embedding Layer
Feed data into NN classifier
Test the classifier
test data

Figure 3.1: System Designing

3.2 Crosslingual Mapping

3.2.1 The Data Set

As mentioned in Section 2.1.2, the quality of word embeddings directly relies on capturing as much statistical information about word co-occurrences as it is possible. This requires a massive amount of data (Mikolov et al., 2017). Trevett et al. (2019) discussed the advantages of using pre-trained word embeddings, proving that they reduce training time, improve performance of a classifier, and reduce over-fitting. Such aforementioned reasons motivated us to also use pre-trained monolingual embeddings for crosslingual mapping in our experiments. We apply German and French pre-trained word vectors[1] in fastText format obtained by Bojanowski et al. (2017) (see Section 2.1.2 for more information about fastText embeddings). Both embedding sets were trained on Wikipedia articles[2], resulting in ca 2 million words vectors for German and ca 1,5 million for French. The vectors are in dimension of 300.

3.2.2 Training

Word embeddings is a powerful method to obtain semantic information just from reading the data. This statement is a fact that has been proven by various works (e.g. Mikolov et al. (2013c); Barone (2016); Duong et al. (2016)). The goal of this thesis is to investigate if word embeddings gained from data in one language can not only learn semantics rules, but also transfer them into another language. Basically, if the model sees a new customer review in French, yet it has never been fed with French data, and the meaning of the review is close to the one it has been trained on in German, the prediction of the model has to be based on the knowledge received from similar German data.

To test the performance of crosslingual word embeddings we first learn a mapping between two sets of individually trained monolingual word vectors, based on the approach introduced by Conneau et al. (2017) (see Section 2.1.3 for a detailed introduction to the approach).

As mentioned in the previous section, we use German and French monolingual embeddings of dimension 300. To address the first research question, we aligned embeddings two times: first with German as a source and then as a target language. The output of the alignment are two sets of embeddings: aligned ("fake") source vectors and unchanged target vectors. Rare words embeddings are usually of less quality than the frequent words embeddings (Luong et al., 2015), that is why the model uses only 200 000 most frequent words from each embedding set for alignment.

[1]word vectors downloaded from https://fasttext.cc/docs/en/pretrained-vectors.html
[2]https://www.wikipedia.org/

The first step of the alignment is adversarial learning, which is basically a one-to-one game of two models - encoder (E) and discriminator (D). The task of D is to predict the language of an embedding. Following Conneau et al. (2017), we use a multilayer perceptron with two hidden layers and ReLU activation function for it. In return E aims to fool the discriminator, pushing the word embeddings as close to each other as it is possible. At every iteration step, E and D are trained sequentially with stochastic gradient (batch size=32), aiming to minimize their loss functions (see Equations 2.17 and 2.18). Every time the unsupervised validation criterion shrinks, the learning rate is divided by two.

The second step seeks to improve the results of adversarial training, applying Procrustes algorithm. The algorithm aims to find out the optimal mapping, using the most frequent words of both embedding sets. To select the best pairs Conneau et al. (2017) introduced a distant metric called cross-domain similarity local scaling (CSLS). It takes into account only those pairs of words that are the nearest neighbours of each other. Moreover, CSLS helps to enlarge the space with high word density, preventing hubs from being as close to all words but their nearest neighbour as they would be otherwise. It results in a smaller but more accurate generated dictionary.

$$CSLS(\overrightarrow{w}_s^*, \overrightarrow{w}_t) = 2\cos(\overrightarrow{w}_s^*, \overrightarrow{w}_t) - r_T(\overrightarrow{w}_s^*) - r_S(\overrightarrow{w}_t), \tag{3.1}$$

where $r_S(\overrightarrow{w}_t)$ is the mean similarity of a target word w_t to its neighbours in the source language. Similarly, $r_T(\overrightarrow{w}_s^*)$ is the mean similarity of a source word to its target neighborhood.

During the alignment we followed step-by-step guide and code[3] provided by Conneau et al. (2017). The results of the crosslingual mapping are discussed in the next chapter.

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

3.3 Evaluation Tasks

3.3.1 The Data Sets

Text Categorization and Sentiment Analysis

For text categorization we use German and French subsets of the Crosslingual Sentiment Dataset (CSD) compiled by Prettenhofer and Stein (2010). The data set originally consists of Amazon customer reviews (obtained in 2009) for three categories: books, music and dvds, in three languages - German, French and Japanese. To avoid the influence of imbalance on the model, Prettenhofer and Stein (2010) balanced the data set, which resulted in 12000 labeled reviews for each language (4000 reviews for each category). Each review has a different length with maximum of 1792 words. We provide example reviews from a German subset[4]:

dvd: *Die DVD ist völlig unnütz und enthält keine relevanten Informationen, die bei einer Reise nach Johannesburg von irgendwelcher Bedeutung wären*

music: *The Cure haben hier eine richtig gute Platte abgeliefert. Es ist mir ein Rätsel, wie man das anders sehen kann.*

books: *Praktische Philosophie: ein Theorie-Klassiker der Frauen-Bewegung. Fundiert und solide recherchiert, lebendig geschrieben, der Inhalt ist packend und erschütternd.*

Moreover, the data set is provided with a sentiment rating (6000 positive and 6000 negative reviews). This fact motivated us to choose it for a sentiment analysis task as well. There are two examples from a French subset below[4]:

positive: *Une de mes séries préférées lors de mon adolescence que j'ai plaisir à revoir. Quel plaisir on prend à observer le contraste et la tumultueuse cohabitation entre les méthodes anglaises et new yorkaises... Plus, pas mal d'action, pas mal fichue... Bonne série. Cependant la série ayant 25 ans, on aurait pu améliorer l'image. Parce que là c'est qualité K7 vidéo...*

negative: *Au bout d'un quart d'heure seulement de visionnement, le film apparaît déjà décousu et déjanté, et fait de lieux communs aussi éculés que*

[4]the style and spellings are original

traités de manière outrée. On peut s'abstenir de l'achat, et se contenter de vérifier mes dires lors d'une programmation à la télévision.

Humour Detection

All previous research into semantic humour detection relied on the collection of textual data - jokes, one-liners, anecdotes. Purandare and Litman (2006) and Acosta (2016) studied the influence of acoustic features for humour detection on acted data - dialogues from a comedy show "Friends" and the transcription of Ted Talks, using the audience laughter as a label for "humorous". In our work we experiment with real dialogue data, as it offers natural expression of human emotions.

The DUEL corpus of a natural face-to-face dialogue (Hough et al., 2016) is a multilingual corpus (German, French and Mandarin Chinese) that consists of audio, video and body tracking data and is transcribed and annotated for laughter, exclamations and disfluences. Same as for the previous classification tasks, we use German and French subsets of the corpus in this thesis.

The speech data selected for this work consists of 10 dialogue recordings per each language. All participants were native speakers. The experiment lasted around 45 minutes for each pair. The subjects were given three development scenarios. At first, participants discussed how they would decorate and furnish their shared apartment if they were given 500 000 Euros. Then they were suggested to write a scene for a film script, in which something odd or ridiculous happens to the main character. And the last task was to participate in a role-play interview. One of the pair, playing the role of a border officer, had to survey a traveller during the border control. The task was complicated by the suggested personal history for each character (e.g. the traveller carries illegal substances, or the characters are related).

Each dialogue was orthographically transcribed and segmented in Praat (Boersma et al., 2002), based on a four-tier approach (Hough et al., 2016) and annotated for laughter, exclamations and speech disfluencies (silent pauses, filled pauses and editing turns, lengthening, repairs, restarts and abandoned utterances (cut-offs)), following the light-weight dialogue mark-up protocol (Hough et al., 2015). The examples of transcription and annotation are provided below:

- **German:**

A: ($< p =$""aber"">a$-$ $< /p >$ + aber) so diesen ganzen ($< p =$""modischen"">mo$-$ $< /p >$ + modischen) Einrichtungskram so das meiste musst du sowieso machen. Weil das bei mir <laughter> dann immer total kahl und kalt aussieht < /laughter>

B: <laughter> < /laughter>

- **French:**

A: {F euh::} on a pas discuté de quoi encore?

B: <laughter> c'était obligé que tu dises ça < /laughter>

3.3.2 Data Pre-processing

Raw data is often distorted and unreliable, and values may be missing. Using such data in modeling can lead to incorrect results, that is why when working with real world textual data the step of its pre-processing is mandatory.

Both textual corpora we use for text classification are multitasking and are provided in XML (CSD) and TextGrid (DUEL) formats. So the first step is to fish out the information relevant for the thesis: pure text and label, omitting other data (e.g. audio, onset and offset time data, metadata). This is followed by tokenizing the data and removing any punctuation and annotation marks (after labeling) such as "<laughter>", "!", ":)", "{F euh:: }", etc (Chollet et al., 2015). Then the tokens are lowercased and converted into numerical vectors. We add an extra word "OOV" with an all zero vector to the word embeddings for those tokens, that do not appear in the embedding vocabulary.

The authors of CSD label the review categories as *"Bücher", "DVD"* and *"Musik"* for a German subset, and *"Livres", "DVD&amp; Blu-ray", "Musique"* for a French subset, hence we first unify the labels into a single form - "books", "dvd" and "music". As mentioned in Section 2.2.1, multiclass classification formally consists of building a binary classifier per each class. To transform the multiclass categories into binary labels we use a *LabelBinarizer* from a scikit-learn library (Pedregosa et al., 2011). Moreover, the data set is labeled with a rating of 1, 2, 4 and 5 stars (the neutral reviews with a rating of 3 stars were removed by the authors of the data set).

Following Blitzer et al. (2007) and Prettenhofer and Stein (2010), we label reviews with 4 and 5 stars as 1 (positive), and reviews with 1 and 2 stars as 0 (negative).

The dialogues of DUEL are marked with speaker turn boundaries. The first idea of labeling humorous and non-humorous turns was pretty straightforward: a turn including a laughter (<laughter>, </laughter> or <laughterOffset/>) mark is labeled 1 (funny), and all the rest are 0 (not funny). But the closer look at the dialogues raised a question about its relevance. For example:

1. A: Meins, glaube ich auch so ein latte-machiato ton oder weiß.

   B: <laughter>

2. A: Le premier c'était une histoire qui c'est vite fini et le deuxième est mort et donc je me suis remarié.

   B: <laughter>

Thus, a new simple ruler-based algorithm for labeling humorous turns was developed. First, following the example above, if a speaker turn is followed by laughter of another speaker only, this turn is labeled as humorous, and the answer turn is removed from the data set. Secondly, if a speaker turn without a laughter mark is followed by the short answer turn with a laughter mark, they build one turn and this turn is marked as humorous:

1. A and B are talking about designing their common flat, A forgot about the bed for B:

   A: Ich denke du besuchst doch nur uns kurz...

   B: <laughter> ohne Anmeldung </laughter>

2. A and B discuss the design of their living room:

   A: Donc il faut imaginer un salon mais genre avec des canapés assez grands ou style canapés lits.

   B: <laughter> Grave <laughter>

In addition, we label the rest of the sentences with a laughter mark with 1:

1. A: Jonas macht sich Sorgen weil Ikesh seit <laughter> drei Stunden nicht mehr bei Whatsapp online war.

2. A: Sauf que sa grand-mère est rentrée à ce moment là pendant: <laughter> son coït </laughter>.

After labeling, we got a total of 11622 speaker turns of which 1413 are humorous and 10209 are non-humorous for the German subset, and a total of 7743 turns with 500 humorous and 7243 non-humorous for the French subset.

3.4 Training and Testing

To evaluate the quality of the crosslingual words embeddings we chose three semantically challenging text classification tasks (see Section 2.2.1 for a detailed introduction). Both data sets we used for classification are labeled data, so we apply a supervised learning algorithm (SLA) to them. The input of SLA is always a pair consisting of an input text (features) and desired output label. Thus, the input of Crosslingual Sentiment Dataset is a set of customer reviews (the vectors of the reviews) and a set of labels - a category of a review (books, music or dvd) or its sentiment (positive or negative). The input of DUEL data set in turn is a set of speaker turns and their humour labels.

The aim of the thesis is to investigate the influence of crosslingual word embeddings on the performance of text classifiers. To solve this, we implement Convolutional Neural Network for topic classification and sentiment analysis tasks, and both Convolutional and Long Short-term Memory NNs for humour recognition.

For building the NN models we use Keras API (Chollet et al., 2015). It is an open-source neural network library written in Python that aims to speed up and ease the developing of NNs. The final models are illustrated in Figure 3.2 and Figure 3.4.

We started building the model with a Keras' Sequential layer and added a word embedding layer, a Convolutional layer, a Max-pooling layer to reduce the number of parameters and to generalize the result of a CNN layer. To reshape the matrix into a vector, a Flatten layer is then applied and fed into

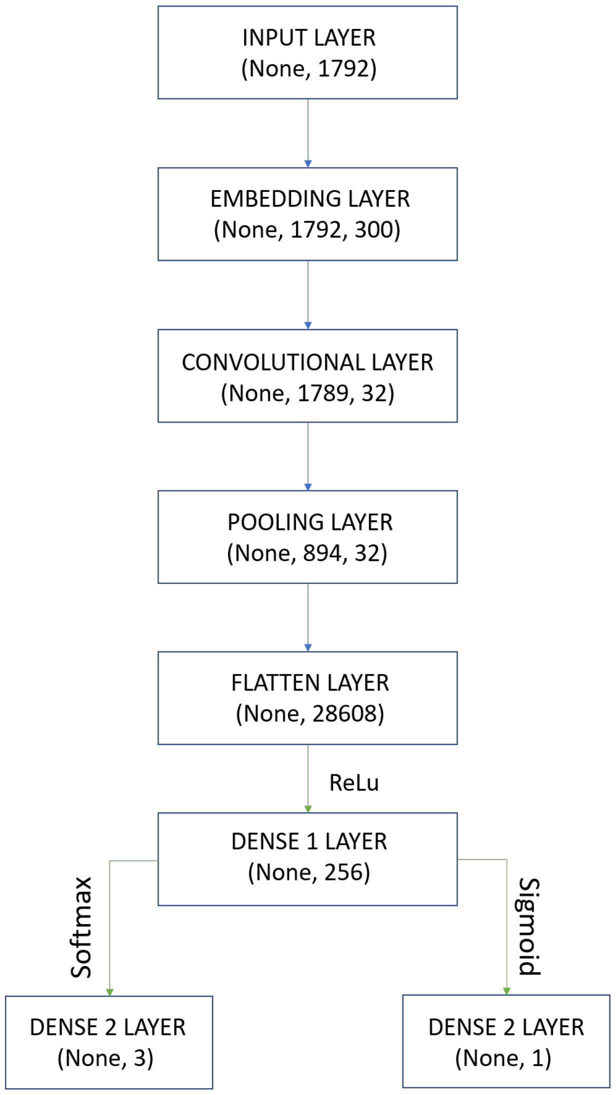


Figure 3.2: The layer architecture of CNN model. The outputs of every layer are shown in parentheses. The length of sequences is normalized by the longest review - 1792 words.

two Fully Collected Dense Layers. The first Dense layer has 256 Nodes and is activated by a ReLU function:

$$F(x) = max(0, x) \tag{3.2}$$

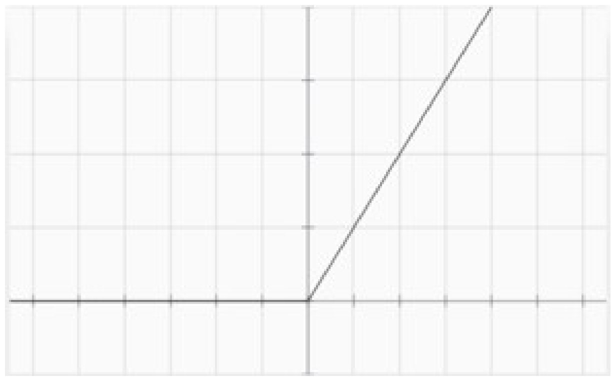


Figure 3.3: The rectified linear activation function (ReLu).

However, ReLu looks (see Figure 3.3) and behaves like a linear function, it is such only for the values greater than zero. It is useful, because having the properties of linear models, ReLu is well generalized and optimized. Moreover, because of its half-linearity, ReLu gives the benefit to activate not all the neurons during the training, but just half of them, which makes the computation efficient and fast.

The second Dense Layer is the output layer. For the binary humour detection and sentiment analysis we use a sigmoid activation function:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.3}$$

Sigmoid models the probability of each of two classes as bernoulli distribution.

Softmax is the extension of a sigmoid function, adapted for multiclass classification. It models the probability of each category as multinominal distribution:

$$f(x_i) = \frac{e^{x_i}}{\sum\limits_{k=1}^{K} e^{x_k}}, \tag{3.4}$$

where K is the number of categories. For the text categorization task there are three labels to predict. The reviews in the data sets are single-labeled, and those labels are independent from each other, so we use a soft-max function at the output layer.

A loss function measures the performance of the model. We apply the categorical cross-entropy loss for text categorization, and the binary cross-entropy loss for binary classification tasks. The output of a loss function is the probability value. A loss function increases if the predicted probability diverges from the actual label.

$$CE = -\sum_{k=1}^{K} t_k \log(f(x_k)), \tag{3.5}$$

Detecting humour in a dialogue differs from its prediction in one-liners or tweets: a dialogue segment is often considered funny only in relation to the dialogue context and the previous speaker turns. That is why, we build a Long Short-Term Memory to model the sequential context of dialogues.

Our LSTM model is formed with an embedding layer followed by three LSTM layers. After some experimentation, we decided to stack LSTM layers for more complexity of the network. If the input of a LSTM layer is the output of a previous LSTM layer, a more sophisticated feature representation is created.

LSTMs are then followed by a flatten layer with a ReLu activation function, and a dropout layer. The DUEL data set is relatively small, and training large NNs can result in data overfitting. Dropout prevents it by randomly dropping units from the neural network during training (Srivastava et al., 2014). Like CNN the LSTM model is fit using the efficient and fast ADAM optimization algorithm (Kingma and Ba, 2014) and a sigmoid loss function.

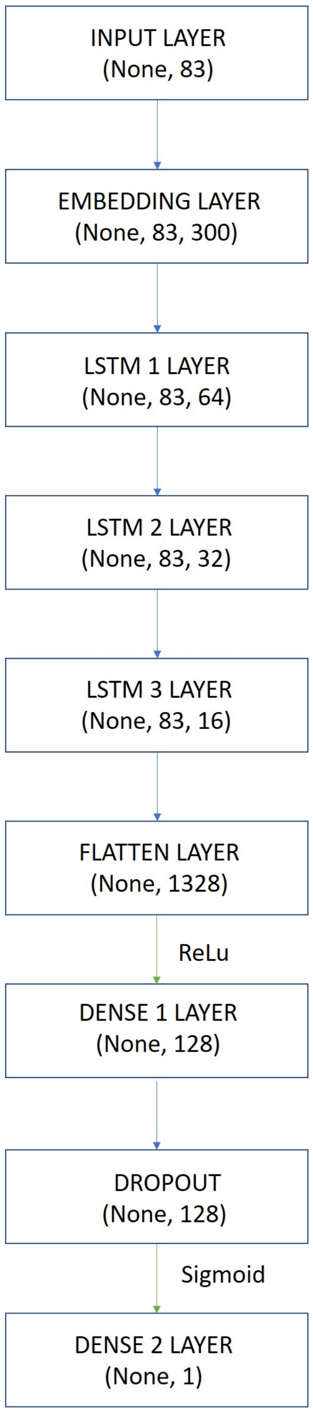


Figure 3.4: The layer architecture of the LSTM model. The outputs of every layer are shown in parentheses. The length of sequences is normalized by the longest speaker turn - 83 words.

Chapter 4

System Evaluation

4.1 Evaluation Methods

To test the performance of any classifier, we set the classifier on the test samples and correlate its prediction of sample classes with ”golden” labels - pre-known classification answers. But in order to make a decision which model copes with the task better, a numerical metric of its quality is required. Most of the metrics are derived from the confusion matrix:

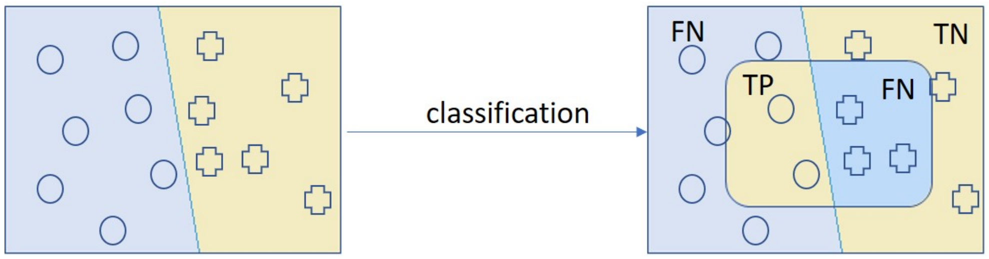


Figure 4.1: The visualization of confusion matrix for a binary classification task. There are two classes: circles and crosses. A small rectangle is a classifier that detects the samples inside it as circles and the samples outside it as crosses. The regions with yellow background depict correctly classified samples, the regions with blue background - mistakes of the classifier.

Confusion matrix contains information about how many times the system makes the right and the wrong decision on the samples of a given class. Namely in Figure 4.1:

true positive (TP) are circles predicted as circles;

true negative (TN) are not circles predicted as not circles;

false positive (FP) are not circles predicted as circles;

false negative (FN) are circles predicted as not circles.

4.1.1 Accuracy

Accuracy is the share of test samples, which the classifier made the right decision on.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{4.1}$$

Accuracy is the most obvious and simple solution. However, this metric has one feature that needs to be considered: it assigns the same weight to all samples, which may not be correct, if the distribution of samples in the training set is strongly shifted towards one or several classes. In this case, the classifier has more information on these classes and, accordingly, it will make more adequate decisions within these categories. In practice, this leads to high overall accuracy, but at the same time the classifier works very poorly within "weaker" classes (Provost et al., 1998).

For text categorization and sentiment analysis tasks we train the model on a specially prepared, balanced corpus of samples. That is why accuracy is a good choice for metric selection and is used as a major metrics for evaluation.

The DUEL corpus we use for humour detection task is unbalanced. Balancing it manually can result in losing the relevant information required for qualitative classification. Another solution is to change the metric of the formal quality assessment.

4.1.2 AUC

'AUC' stands for 'area under the curve'. In this thesis we specify 'curve' as a receiving operation characteristic curve (ROC curve), which is most often used to represent the results of binary classification in machine learning. The AUC of a classifier represents the probability of the classifier to randomly pick a positive example or a negative example. AUC measures the entire two-dimensional area under the curve (calculates the integral) from (0,0) to (1,1). The ideal ROC curve is the curve passing through the point (0, 1), the area under it is 1. The worst is the ROC curve passing through the point (1, 0), the area under it is 0. The model that does not learn anything is represented at (0,5, 0,5).

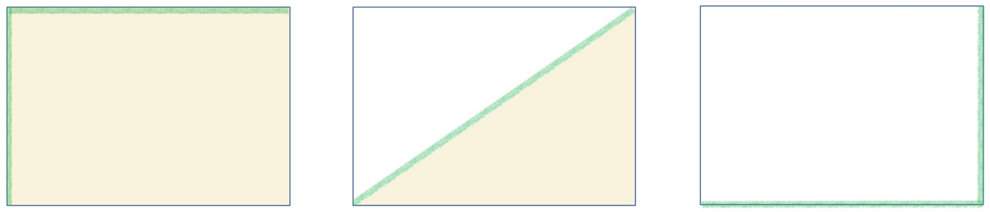


Figure 4.2: ROC-curves of the best (AUC=1), the random (AUC=0.5) and the worst (AUC=0) models

The ROC curve shows the dependence of true positive samples (TP) on false positive samples (FP), assuming that the classifier has a certain parameter called a cut-off value. Varying this parameter results in a different partition of two classes, and, hence, a different amount of TP and FP samples. For example, the lowering of the cutoff value leads to the following: the system classifies more examples as positive, while increasing both FP and TP.

$$TPR = \frac{TP}{TP + FN} \tag{4.2}$$

$$FPR = \frac{FP}{TN + FP} \tag{4.3}$$

For a ROC curve, True Positive Rate (TPR) (Equation 4.2) and False Positive Rate (FPR) (Equation 4.3) are computed with many different cut-off values and plotted on a single graph with FPR values on the abscissa and TPR values on the ordinate.

As TPR and FPR are calculated for each class separately, the ROC curve is invariant to class ratios and is a good metric to evaluate the performance of a classifier even for an unbalanced data set (Provost et al., 1998).

4.1.3 Precision, Recall and F1-score

Davis and Goadrich (2006) and Drummond and Holte (2004) argued with Provost et al. (1998) stating that the ROC curve provides "an overly optimistic view of an algorithm's performance" (Davis and Goadrich, 2006). They experimentally proved that in some cases of highly unbalanced data sets precision and recall give a more informative picture of the model's performance.

Precision and recall are metrics used for evaluation of an algorithm's performance on each of the classes separately. Precision can be interpreted as the proportion of objects that the classifier calls positive and which are actually positive, and recall shows how many objects of the positive class among all objects of the positive class the model has found.

$$Precision = \frac{TP}{TP + FP} \tag{4.4}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.5}$$

Recall demonstrates the ability of an algorithm to detect a given class in general and precision shows the ability to distinguish this class from other classes. Unlike accuracy precision and recall do not depend on the class ratios and are therefore useful in conditions of unbalanced samples.

The F1-score is an evaluation method which considers both precision and recall to compute the score. It tends to zero if precision or recall tend to zero. It is often called the weighted average, or the harmonic mean of precision and recall:

$$F1 = \frac{(precision^{-1} + recall^{-1})^{-1}}{2} = 2 \times \frac{precision \times recall}{precision + recall} \tag{4.6}$$

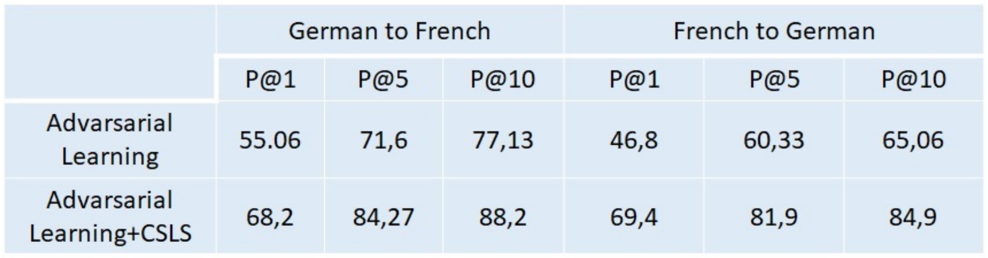
We evaluate each classifier using accuracy, and precision, recall and F1-score of each class separately. Moreover, for both binary classification tasks we plot an AUC ROC curve.

4.2 Results and Discussion

4.2.1 Crosslingual mapping: Word Translation

The approach we use to map monolingual word embeddings into a shared vector space is unsupervised, but to investigate the effectiveness of the mapping, in their work Conneau et al. both generated their own dictionaries, which handle polysemy, and used dictionaries presented in Dinu et al. (2014). These dictionaries are then applied as golden standards to evaluate how often the model finds the correct translation of a test word.

We report results for a German-French language pair on the dictionaries provided by Conneau et al. (2017)[1]. Table 4.1 shows the average precision P@ for 1, 5 and 10 words.



	German to French			French to German		
	P@1	P@5	P@10	P@1	P@5	P@10
Advarsarial Learning	55.06	71,6	77,13	46,8	60,33	65,06
Advarsarial Learning+CSLS	68,2	84,27	88,2	69,4	81,9	84,9

Table 4.1: Word translation task results for $P@1, P@5, P@10$. Comparison of results directly after adversarial training and after adding the refinement Procrustes step.

Precision at k $P@k$ stands for the k number of relevant translations of a source word. For example, there are 5 suggested translations of a word, where only the first and the fourth were found in evaluation dictionaries. The precision of every translation is then counted separately ($p@1 = 1/1 = 1$; $p@4 = 2/4 = 0,5$; $p@2,3,5 = 0$), and summed up for an average precision score ($P@5 = \frac{1+0+0+0,5+0}{2} = 0,75$). In our analysis we observe 1500 source words and 200 000 translations.

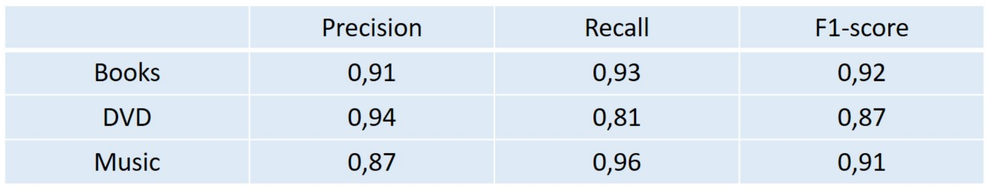
Table 4.1 demonstrates the impact of the Procrustes algorithm on crosslingual mapping. However, the results obtained just with an adversarial learning approach are at a good level, the improvement of the performance is significant, when adding the refinement step.

[1]https://github.com/facebookresearch/MUSE#ground-truth-bilingual-dictionaries

The results for German-French and French-German mapping are essentially the same, therefore it proves, that the change of the source language in the pair of languages does not have a significant impact on the quality of alignment. Thus, we evaluate only German-French mapping on all text classification tasks.

4.2.2 Text Categorization

We test the performance of crosslingual word embeddings using the previously described German and French data sets, and report results when training in German and testing in French in Table 4.2.



	Precision	Recall	F1-score
Books	0,91	0,93	0,92
DVD	0,94	0,81	0,87
Music	0,87	0,96	0,91

Table 4.2: Precision, recall and F1-score of each category in our data set. The model is trained on 10 000 German text samples (2 000 are used as a validation set) and tested on 12 000 French samples.

F1-scores reveal that the classifier is able to both detect the classes in the data set and distinguish the classes from each other successfully. Precision and recall demonstrate that it is extremely demanding when classifying reviews into category 'DVD', which sometimes leads to TPs' mislabeling.

We obtain **90,05% categorical accuracy** when training in German and testing in French. While a lot of studies focused on classifying the reviews from the same data set crosslingualy, none of them treat German as a source language (more information about the works in Section 2.2.2). Schwenk and Li (2018) trained a classifier on a small corpus of 1 000 news stories, reaching 71.55% accuracy for a German-French pair, however the input data does not allow the comparison with our results.

We analyze incorrectly labelled data and report that most of the samples are either refer to a 'neutral' category, for example: *"Très content du produit, délai de livraison respecter. Merci."* (I am very happy with the product, delivery was on time. Thank you.), or to multiple categories: *"Il n'y pas d'histoire dans le film. Seule la musique est bonne, qui sauve ce navet d'un*

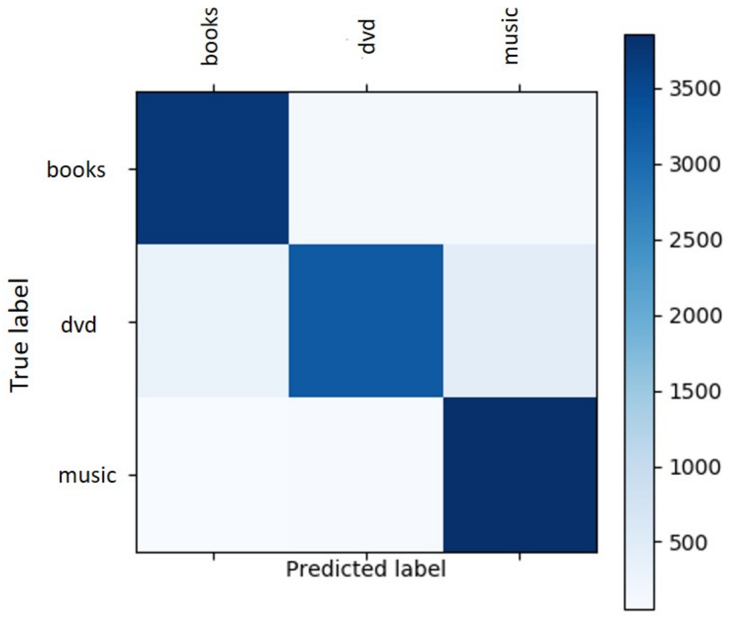
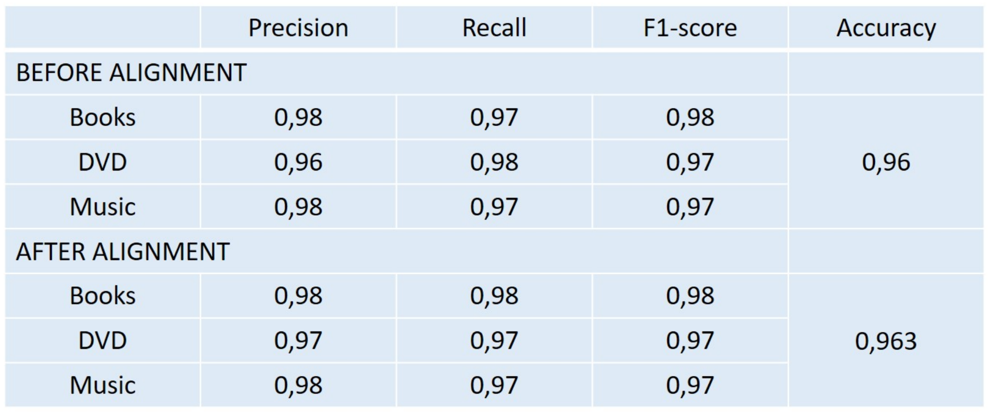


Figure 4.3: Visualization of the results of the German-French text categorization task for each category.



	Precision	Recall	F1-score	Accuracy
BEFORE ALIGNMENT				
Books	0,98	0,97	0,98	0,96
DVD	0,96	0,98	0,97	
Music	0,98	0,97	0,97	
AFTER ALIGNMENT				
Books	0,98	0,98	0,98	0,963
DVD	0,97	0,97	0,97	
Music	0,98	0,97	0,97	

Table 4.3: Results of monolingual text categorization using original FastText German embeddings and German embeddings aligned into a shared German-French space. The models are trained on 7 000 German text samples (1 000 are used as a validation set) and tested on 4 000 German samples.

plongeon dans les abîmes." (There is no story in the film. The only good thing is music, which saves it from falling into an abyss.).

To track the influence of crosslingual mapping on the performance of the classifier, we compare the results of monolingual classification using German embeddings before and after alignment. It is predictable, that the accuracy of monolingual classification is significantly higher than crosslingual. However, based on Table 4.3, it is clear that crosslingual mapping does not influence semantic relations between words within one language.

To better understand how well the embeddings pick up crosslingual connections between words and to compare their quality with monolingual relations, we train and test the model on the French data. Although Table 4.4 demonstrates the significant increase of the performance when trained monolingually, we take into account the possibility to operate with substantial training data that must not be the case for many under-resourced languages.

4.2.3 Sentiment Analysis

The results of sentiment analysis across languages are presented in Table 4.5. From the results it can be noted that the algorithm performs better when classifying positive reviews. We follow Dodds et al. (2015) with the assumption that this is due to the fact that people use a larger number

	Precision	Recall	F1-score	Accuracy
Books	0,97	0,98	0,98	0,97
DVD	0,97	0,95	0,96	
Music	0,98	0,98	0,98	

Table 4.4: Results of monolingual text categorization using original French word embeddings. The model is trained on 7 000 French text samples (1 000 are used as a validation set) and tested on 4 000 French samples.

of positive than negative words, as naturally ”language possess a universal positivity bias” (Dodds et al., 2015).

	Precision	Recall	F1-score	Accuracy
Negative	0,75	0,61	0,67	0,73
Positive	0,67	0,8	0,73	

Table 4.5: Precision, recall and F1-score of negative and positive reviews in our data set. The model is trained on 10 000 German text samples (2 000 are used as a validation set) and tested on 12 000 French samples.

In Figure 4.4 the area under the blue line corresponds to AUC. The dashed line is the random predictor (baseline) used to evaluate the usefulness of the model. Although the model does not distinguish classes ideally, it definetely has some predictive power. Like in the previous section direct comparison of the results between ours and other systems is not possible, because other experiments were done either on very different data sets, or by using different language pairs.

As well as for text categorization task we notice that crosslingual mapping neither weakens nor strengthens correlations between words within one language (Table 4.6).

Table 4.7 provides the results of monolingual sentiment analysis of the French reviews. Interestingly, the model shows nearly the same amount of accuracy points decrease as in text categorization task (TC: 97 $\rightarrow$ 90, SA: 81 $\rightarrow$ 73), when trained crosslingually.

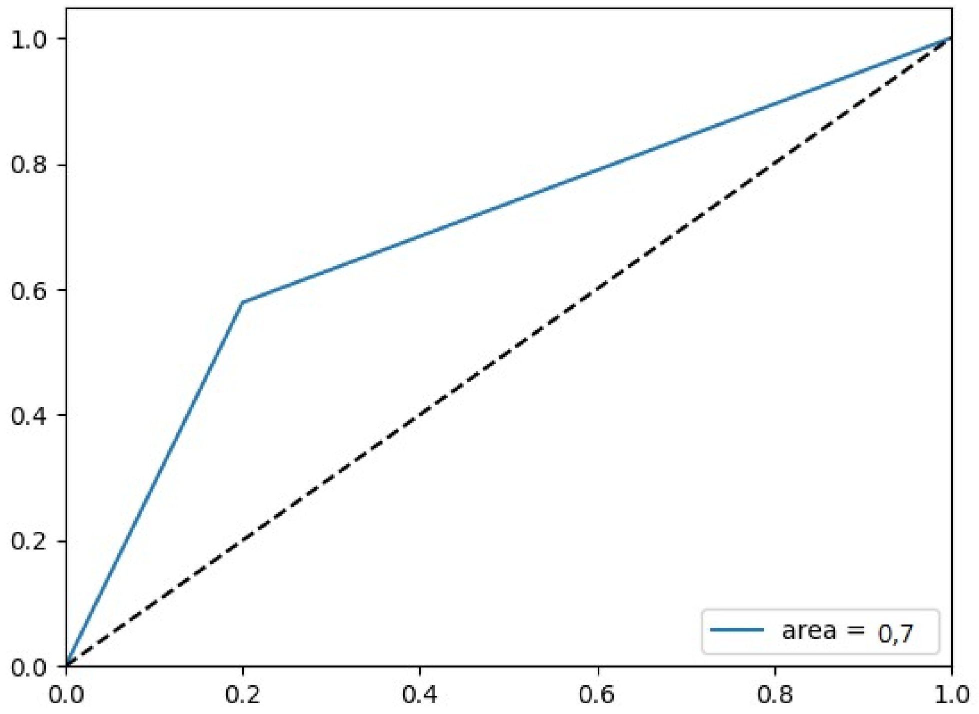
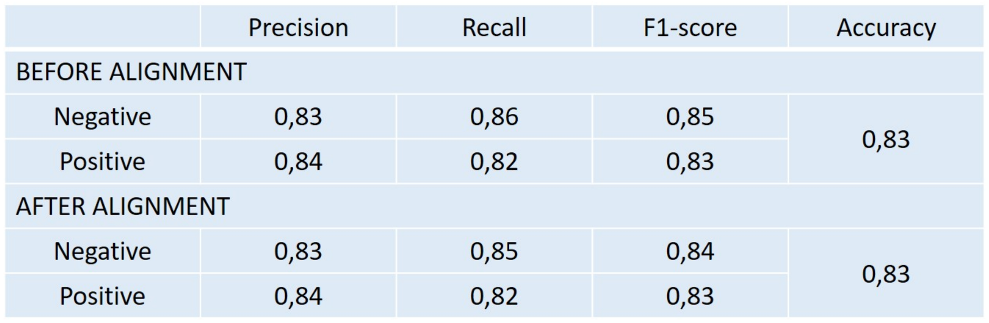


Figure 4.4: Area under the curve of the Receiver Operating Characteristic (AUCROC) of the crosslingual sentiment analysis task



	Precision	Recall	F1-score	Accuracy
BEFORE ALIGNMENT				
Negative	0,83	0,86	0,85	0,83
Positive	0,84	0,82	0,83	
AFTER ALIGNMENT				
Negative	0,83	0,85	0,84	0,83
Positive	0,84	0,82	0,83	

Table 4.6: Results of monolingual sentiment analysis using original FastText German embeddings and German embeddings aligned into a shared German-French space. The models are trained on 7 000 German text samples (1 000 are used as a validation set) and tested on 4 000 German samples.

	Precision	Recall	F1-score	Accuracy
Negative	0,80	0,86	0,83	0,81
Positive	0,79	0,83	0,8	

Table 4.7: Results of monolingual sentiment analysis using original French word embeddings. The model is trained on 7 000 French text samples (1 000 are used as a validation set) and tested on 4 000 French samples.

4.2.4 Humour Detection

This section presents the results of a crosslingual humour detection experiment using CNN and LSTM classifiers (see Table 4.8 and Figure 4.5). Although LSTM outperforms CNN, both classifiers show poor performance not reaching the accuracy baseline ($baseline = 0,935$). However, the reason of such performance does not lie in the inability of the model to transfer the semantics across languages: we also test it on monolingual data, with no success (see Table 4.9 and Table 4.10).

		Precision	Recall	F1-score	Accuracy
CNN	Humorous	0,12	0,24	0,16	0.83
	Non-humorous	0,94	0,87	0,91	
LSTM	Humorous	0,15	0,3	0,2	0.85
	Non-humorous	0,95	0,89	0,92	

Table 4.8: Precision, recall and F1-score of humorous and non-humorous speaker turns of the DUEL data set. The model is trained on 10 000 German sentence turns (1 622 are used as a validation set) and tested on 7 743 samples in French.

We choose natural human dialogue for detecting humour, because we find it more challenging and useful for future applications. However, there are several probable causes of the inefficiency of our model. First, the ratio between non-humorous and humorous speaker turns is too big, which leads to the model being biased. As the data set is already scarce, we did not consider undersampling to solve this. We used scikit-learn func-

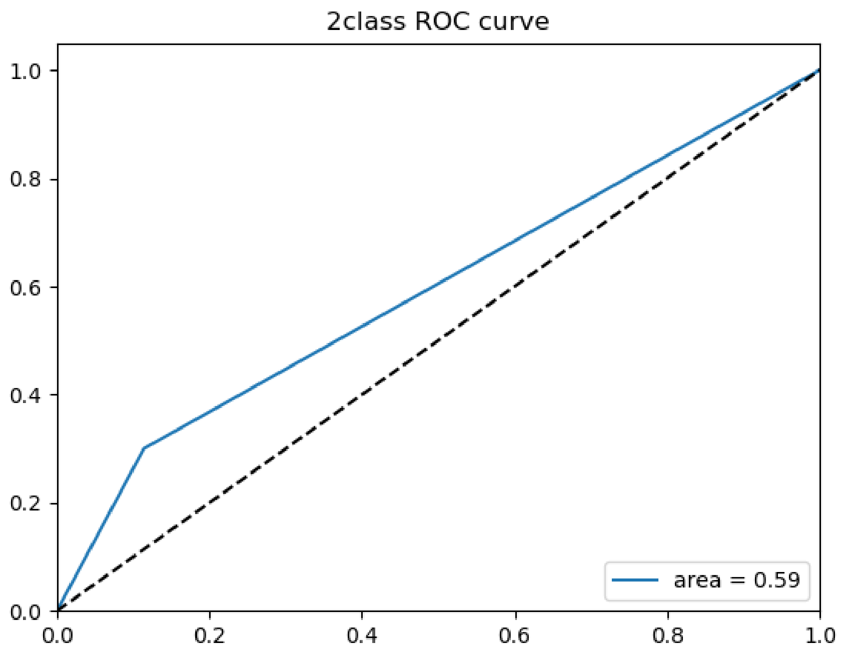
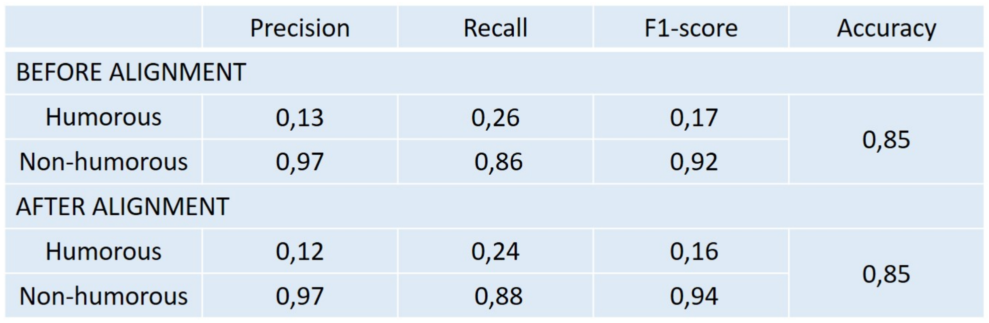


Figure 4.5: Area under the curve of the Receiver Operating Characteristic (AUCROC) of the crosslingual humour detection task



	Precision	Recall	F1-score	Accuracy
BEFORE ALIGNMENT				
Humorous	0,13	0,26	0,17	0,85
Non-humorous	0,97	0,86	0,92	
AFTER ALIGNMENT				
Humorous	0,12	0,24	0,16	0,85
Non-humorous	0,97	0,88	0,94	

Table 4.9: Results of monolingual humour detection (LSTM) using original FastText German embeddings and German embeddings aligned into a shared German-French space. The models are trained on 7 000 German text samples (1 000 are used as a validation set) and tested on 4 000 German samples.

tion *compute_class_weight* (Pedregosa et al., 2011) to give higher weight to the humorous class. It led to a better performance on the minority class, but to a worse performance of the model itself. Secondly, in this and some other works on humour computation, laughter is treated as a label for a humorous sample. However, analyzing the content of the DUEL dialogues, namely the turns that provoke laughter, we come to the conclusion that laughter does not always mean humour. In their work Aragón et al. (2015) noted that sometimes "laughter serves a self-regulation function", that helps people relax when they feel nervous or uncomfortable. Edmonds and Miller (2009) discussed different causes that provoke laughter. One of them is etiquette, which we use to help others feel respect and belonging. That is why, we assume that deeper investigation and labeling is required to compute humour in a dialogue for a better performance.

	Precision	Recall	F1-score	Accuracy
Humorous	0,11	0,22	0,14	0,81
Non-humorous	0,96	0,84	0,89	

Table 4.10: Results of monolingual humour detection (LSTM) when using original French word embeddings. The model is trained on 7 000 French text samples (1 000 are used as a validation set) and tested on 4 000 French samples.

Like for the previous classification tasks we investigate the impact of crosslingual mapping on the monolingual relations (Table 4.9) and observe the performance difference between crosslingual and monolingual laughter detection models (Table 4.10). As already mentioned, the model does not recognize humour even when trained and tested on data in one language. Another reason that may have caused such results is that humour detection in natural dialogues cannot be computed based just on semantics, and requires some non-verbal features and extra-lingual information for a successful performance.

The next chapter introduces conclusions of the work that has been done as well as discussions about possibilities for performing future work.

Chapter 5

Conclusions

5.1 Conclusion

Transferring semantic information across languages has been under research since the 2010s and is still in a developing phase, as more and more different models for learning crosslingual mapping are proposed in the academia. This thesis investigated the existing algorithms for obtaining crosslingual word embeddings: it gave a comprehensive introduction into the topic, showed differences and similarities between the approaches, pointed out their weaknesses and strengths. According to the literature review, it was found that the majority of existing works used English data as source data for learning crosslingual mapping. Therefore, we decided to go another way: we chose a German-French pair of languages in our experiments.

We applied the crosslingual mapping approach suggested by Conneau et al. (2017) in the thesis. The method requires minimal amount of data for mapping, and thus, can be easily applied for resource-rich-low-resource language transfer in future applications. In the second part of the experiment we explored the embeddings aligned in a shared space on three challenging text classification tasks: multi-class text categorization, sentiment analysis and semantic humour detection in a dialogue. We built a CNN classifier for every task, and, additionally an LSTM classifier for a humour recognition model. The classifiers of the first two tasks showed sufficient results on crosslingual text categorization and sentiment analysis, obtaining 90,05% accuracy when labeling the topic of the customer reviews, and 73% when predicting the sentiment of the reviews. However, the models turned out to be useless when predicting humour in dialogue speaker turns.

Additionally, this study has shown that aligning source embeddings into a shared vector space does not influence the quality of these "fake" embeddings. The monolingual connections between words stay the same after crosslingual mapping.

Finally, we observed the performance of the classifiers on French data, comparing the results between the classifier trained and tested on a French subset, and the classifier trained in German and tested in French. Although the results were predictable, and classification within one language achieved a better performance, it is necessary to pay attention to the fact that we were able to operate with structured and labeled training data that are scarce or non-existent for less resourced languages than French.

All in all, the experiments have shown that crosslingual word embeddings are able to transfer some semantic information across languages and are extremely useful when lacking the training data. As for more semantically challenging tasks such as humour detection, the further experimentation is required to improve the performance. It is discussed in the next section.

5.2 Future Work

For future work it could be worthwhile to test the crosslingual classifiers on more challenging real-life data sets, such as Twitter or Instagram posts. It is interesting, if semantics can be obtained and transferred across languages from the data sets, which consist of misspellings, slang and emoticons.

Moreover, testing the crosslingual mapping between resourced and under-resourced languages, and comparing the performance of a classifier when trained on a big data set on the resourced language and tested on the under-resourced data, and when trained on scarce data of the under-resourced language and tested on the same language data is required.

As for the humour detection, some non-arbitrary decisions of what humorous turns are, have to be made. Testing the same model on a different balanced dialogue data, preferably, manually labelled for funny remarks, and not for acoustic laughter may be sufficient. Additionally, it can make the contribution to experiment with different semantic features, which are considered to provoke laughter, such as ambiguous words or adult slang.

Bibliography

Acosta, A. D. (2016). Laff-o-tron: Laugh prediction in ted talks. Master's thesis, California Polytechnic State University, San Luis Obispo.

Adel, H. and Schütze, H. (2016). Exploring different dimensions of attention for uncertainty detection. *arXiv preprint arXiv:1612.06549*.

Aragón, O. R., Clark, M. S., Dyer, R. L., and Bargh, J. A. (2015). Dimorphous expressions of positive emotion: Displays of both care and aggression in response to cute stimuli. *Psychological science*, 26(3):259–273.

Artetxe, M., Labaka, G., and Agirre, E. (2016). Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2289–2294.

Artetxe, M., Labaka, G., and Agirre, E. (2017). Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462.

Artetxe, M. and Schwenk, H. (2018). Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *arXiv preprint arXiv:1812.10464*.

Barone, A. V. M. (2016). Towards cross-lingual distributed representations without parallel text trained with adversarial autoencoders. *arXiv preprint arXiv:1608.02996*.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *Journal of machine learning research*, 3(2):1137–1155.

Blitzer, J., Dredze, M., and Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Boersma, P. et al. (2002). Praat, a system for doing phonetics by computer. *Glot international*, 5.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Budanitsky, A. and Hirst, G. (2001). Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and other lexical resources*, volume 2, pages 2–2.

Bullinaria, J. A. and Levy, J. P. (2007). Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526.

Chandar, S. A., Lauly, S., Larochelle, H., Khapra, M. M., Ravindran, B., Raykar, V., and Amrita, S. (2014). An autoencoder approach to learning bilingual word representations. In *Advances in Neural Information Processing Systems*, pages 1853–1861.

Chen, L. and Lee, C. M. (2017). Predicting audience's laughter using convolutional neural network. *arXiv preprint arXiv:1702.02584*.

Chiarello, C., Burgess, C., Richards, L., and Pollock, A. (1990). Semantic and associative priming in the cerebral hemispheres: Some words do, some words don't... sometimes, some places. *Brain and language*, 38(1):75–104.

Chollet, F. et al. (2015). Keras. https://keras.io.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Cohen, W. W. (1995). Learning to classify english text with ilp methods. *Advances in inductive logic programming*, 32:124–143.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(8):2493–2537.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., and Jégou, H. (2017). Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.

Conneau, A., Schwenk, H., Barrault, L., and Lecun, Y. (2016). Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*.

Das, D. and Petrov, S. (2011). Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 600–609. Association for Computational Linguistics.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM.

de Oliveira, L. and Rodrigo, A. L. (2017). Humor detection in yelp reviews.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Dinu, G., Lazaridou, A., and Baroni, M. (2014). Improving zero-shot learning by mitigating the hubness problem. *arXiv preprint arXiv:1412.6568*.

Dodds, P. S., Clark, E. M., Desu, S., Frank, M. R., Reagan, A. J., Williams, J. R., Mitchell, L., Harris, K. D., Kloumann, I. M., Bagrow, J. P., et al. (2015). Human language reveals a universal positivity bias. *Proceedings of the National Academy of Sciences*, 112(8):2389–2394.

Dou, Z.-Y. (2017). Capturing user and product information for document level sentiment analysis with deep memory network. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 521–526.

Drummond, C. and Holte, R. C. (2004). What roc curves can't do (and cost curves can). In *ROCAI*, pages 19–26. Citeseer.

Duong, L., Kanayama, H., Ma, T., Bird, S., and Cohn, T. (2016). Learning crosslingual word embeddings without bilingual corpora. *arXiv preprint arXiv:1606.09403*.

Edmonds, M. and Miller, J. (2009). 10 different types of laughter. https://science.howstuffworks.com/life/inside-the-mind/emotions/5 -types-of-laughter.htm. Accessed: 2019-06-05.

Faruqui, M. and Dyer, C. (2014). Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471.

Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.

Forman, G. and Cohen, I. (2004). Learning from little: Comparison of classifiers given little training. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 161–172. Springer.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. (1983). Human factors and behavioral science: Statistical semantics: Analysis of the potential performance of key-word information systems. *The Bell System Technical Journal*, 62(6):1753–1806.

Gao, H., Hu, J., Wilson, C., Li, Z., Chen, Y., and Zhao, B. Y. (2010). Detecting and characterizing social spam campaigns. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, pages 35–47. ACM.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7(2):155–170.

Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis Lectures on Human Language Technologies*, 10(1):1–309.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Gouws, S., Bengio, Y., and Corrado, G. (2015). Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 748–756, Lille, France. PMLR.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(2):307–361.

Harris, Z. S. (1954). Distributional structure. *Word*, 10(2-3):146–162.

Hayes, P. J., Andersen, P. M., Nirenburg, I. B., and Schmandt, L. M. (1990). Tcs: a shell for content-based text categorization. In *Sixth Conference on Artificial Intelligence for Applications*, pages 320–326. IEEE.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

Hermann, K. M. and Blunsom, P. (2013). Multilingual distributed representations without word alignment. *arXiv preprint arXiv:1312.6173*.

Hindle, D. (1990). Noun classification from predicate-argument structures. In *28th Annual meeting of the Association for Computational Linguistics*.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1).

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Hough, J., de Ruiter, L., Betz, S., and Schlangen, D. (2015). Disfluency and laughter annotation in a light-weight dialogue mark-up protocol. In *The 7th Workshop on Disfluency in Spontaneous Speech (DiSS)*.

Hough, J., Tian, Y., de Ruiter, L., Betz, S., Kousidis, S., Schlangen, D., and Ginzburg, J. (2016). Duel: A multi-lingual multimodal dialogue corpus for disfluency, exclamations and laughter. In *10th edition of the Language Resources and Evaluation Conference*.

James, J. (2016). Data never sleeps 3.0. *Retrieved Syyskuu*, 28:2016.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Johnson, R. and Zhang, T. (2014). Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058*.

Jurafsky, D. and Martin, J. H. (2014). *Speech and language processing*, volume 3. Pearson London.

Kim, P. (2006). The forrester wave: Brand monitoring, q3 2006. *Forrester Wave (white paper)*.

Kim, S.-B., Rim, H.-C., Yook, D., and Lim, H.-S. (2002). Effective methods for improving naive bayes text classifiers. In *Pacific Rim International Conference on Artificial Intelligence*, pages 414–423. Springer.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Klementiev, A., Titov, I., and Bhattarai, B. (2012). Inducing crosslingual distributed representations of words. *Proceedings of COLING 2012*, pages 1459–1474.

Kłopotek, M. A. and Woch, M. (2003). Very large bayesian networks in text classification. In *International Conference on Computational Science*, pages 397–406. Springer.

Lazaridou, A., Dinu, G., and Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 270–280.

Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562.

Lee, H. Y. and Renganathan, H. (2011). Chinese sentiment analysis using maximum entropy. In *Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology*, pages 89–93.

Leopold, E. and Kindermann, J. (2002). Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46(1-3):423–444.

Lewis, D. D. (1992). An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–50. ACM.

Lewis, D. D. and Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In *Third annual symposium on document analysis and information retrieval*, volume 33, pages 81–93.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(6):361–397.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*, volume 2.

Liu, P., Qiu, X., and Huang, X. (2016). Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Liu, Y. and Zhang, M. (2018). Neural network methods for natural language processing.

Lu, A., Wang, W., Bansal, M., Gimpel, K., and Livescu, K. (2015). Deep multilingual correlation for improved word embeddings. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 250–256.

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

Maron, M. E. (1961). Automatic indexing: an experimental inquiry. *Journal of the ACM (JACM)*, 8(3):404–417.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752(1), pages 41–48. Citeseer.

McClulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in neurons activity. *Bulletin of mathematical biophysics*, 5:115–133.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., et al. (2011). Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182.

Mihalcea, R. and Strapparava, C. (2006). Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence*, 22(2):126–142.

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

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. (2017). Advances in pre-training distributed word representations. *arXiv preprint arXiv:1712.09405*.

Mikolov, T., Le, Q. V., and Sutskever, I. (2013b). Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013c). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013d). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Minsky, M. and Papert, S. (1969). An introduction to computational geometry. *Cambridge tiass., HIT*.

Mitchell, T. M. (1996). *Machine Learning*. McGraw-Hill Science, New York, US.

Mogadala, A. and Rettinger, A. (2016). Bilingual word embeddings from parallel and non-parallel corpora for cross-language text classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 692–702.

Ng, A. Y. and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in neural information processing systems*, pages 841–848.

Nigam, K., Lafferty, J., and McCallum, A. (1999). Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1(1), pages 61–67.

Otter, D. W., Medina, J. R., and Kalita, J. K. (2018). A survey of the usages of deep learning in natural language processing. *arXiv preprint arXiv:1807.10854*.

Paatero, P. and Tapper, U. (1994). Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126.

Pang, B., Lee, L., et al. (2008). Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1-2):1–135.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

Pantel, P., Ravichandran, D., and Hovy, E. (2004). Towards terascale knowledge acquisition. In *Proceedings of the 20th international conference on Computational Linguistics*, page 771. Association for Computational Linguistics.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

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

Pham, H., Luong, T., and Manning, C. (2015). Learning distributed representations for multilingual text sequences. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 88–94.

Prettenhofer, P. and Stein, B. (2010). Cross-language text classification using structural correspondence learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 1118–1127.

Provost, F. J., Fawcett, T., Kohavi, R., et al. (1998). The case against accuracy estimation for comparing induction algorithms. In *ICML*, volume 98, pages 445–453.

Purandare, A. and Litman, D. (2006). Humor: Prosody analysis and automatic recognition for f* r* i* e* n* d* s. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 208–215. Association for Computational Linguistics.

Qiao, H. (2015). New svd based initialization strategy for non-negative matrix factorization. *Pattern Recognition Letters*, 63:71–77.

Raskin, V. (1985). *Semantic mechanisms of humor*, volume 24. D. Reidel Publishing Company, Dordrecht, Holland.

Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.

Riloff, E. and Shepherd, J. (1997). A corpus-based approach for building semantic lexicons. *arXiv preprint cmp-lg/9706013*.

Robertson, S. E. and Harding, P. (1984). Probabilistic automatic indexing by learning from human indexers. *Journal of Documentation*, 40(4):264–270.

Rosenblatt, F. (1961). Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY.

Ruch, W. (2002). Computers with a personality? lessons to be learned from studies of the psychology of humor. In *Proceeding of The April Fools Day Workshop on Computational Humor*, pages 57–70.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. (1988). Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1.

Sak, H., Senior, A., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*.

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Sasaki, M. and Kita, K. (1998). Rule-based text categorization using hierarchical categories. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 3, pages 2827–2830. IEEE.

Schwenk, H. and Li, X. (2018). A corpus for multilingual document classification in eight languages. *arXiv preprint arXiv:1805.09821*.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

Sebastiani, F. (2005). Text categorization. In *Encyclopedia of Database Technologies and Applications*, pages 683–687. IGI Global.

Shigeto, Y., Suzuki, I., Hara, K., Shimbo, M., and Matsumoto, Y. (2015). Ridge regression, hubness, and zero-shot learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 135–151. Springer.

Smith, S. L., Turban, D. H., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Socher, R., Biemann, C., and Osswald, R. (2007). Combining contexts in lexicon learning for semantic parsing. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA 2007)*, pages 175–182.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Tang, J., Zhang, Y., Sun, J., Rao, J., Yu, W., Chen, Y., and Fong, A. C. M. (2012). Quantitative study of individual emotional states in social networks. *IEEE Transactions on Affective Computing*, 3(2):132–144.

Trevett, B., Reay, D., and Taylor, N. (2019). The effectiveness of pre-trained code embeddings.

Turney, P. D. (2002). Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 417–424. Association for Computational Linguistics.

Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188.

Wang, P., Qian, Y., Soong, F. K., He, L., and Zhao, H. (2015). Part-of-speech tagging with bidirectional long short-term memory recurrent neural network. *arXiv preprint arXiv:1510.06168*.

Xiao, M. and Guo, Y. (2013). Semi-supervised representation learning for cross-lingual text classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1465–1475.

Xing, C., Wang, D., Liu, C., and Lin, Y. (2015). Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011.

Yang, D., Lavie, A., Dyer, C., and Hovy, E. (2015). Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2367–2376.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90.

Yih, W.-t., Richardson, M., Meek, C., Chang, M.-W., and Suh, J. (2016). The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 201–206.

Zhang, M., Liu, Y., Luan, H., and Sun, M. (2017). Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1959–1970.

Zhang, W., Xu, H., and Wan, W. (2012). Weakness finder: Find product weakness from chinese reviews by using aspects based sentiment analysis. *Expert Systems with Applications*, 39(11):10283–10291.

Zhang, Y., Gaddy, D., Barzilay, R., and Jaakkola, T. (2016). Ten pairs to tag-multilingual pos tagging via coarse mapping between embeddings. In *15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Zhou, X., Wan, X., and Xiao, J. (2016). Cross-lingual sentiment classification with bilingual document representation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1403–1412.