University of West Bohemia
Department of Computer Science And Engineering
Univerzitni 22
306 14 Plzen
Czech Republic

Deep Neural Networks for Selected Natural Language Processing Tasks

PhD Study Report

Ing. Jiří Martínek

Technical Report No. DCSE/TR-2019-02
June, 2019

Abstract

This report presents research in several tasks of the natural language processing, namely optical character recognition, text categorization and dialogue act recognition. The report is focused on modern deep neural network classifiers, which are first introduced theoretically with the support of relevant publications and then they are experimentally verified on suitable datasets. For text categorization and dialogue act recognition tasks, multi- and cross-lingual approaches are presented and evaluated. Since the importance of multi-lingual text processing is significantly growing, it is crucial to be able to process information in multiple languages. Unfortunately, there is often a lack of annotated data in some not very widespread languages, so the cross-lingual approaches are requested. Methods for optical character recognition in historical documents are presented with the main emphasis on classifiers which achieve excellent results in natural language processing tasks. Finally, the aims of the future doctoral thesis are set with regard to the recent research.

Acknowledgements

I would like to express my thanks to my supervisor doc. Ing. Pavel Kral, Ph.D and Ing. Ladislav Lenc, Ph.D for their support, consultations and for valuable advices during my study.

Contents

1 Introduction **1**

2 Deep Learning Survey **2**
2.1 Deep Neural Networks 2
2.1.1 Recursive Neural Network 2
2.1.2 Recurrent Neural Network 4
2.1.3 Convolutional Neural Network 9
2.2 Summary 11

3 Optical Character Recognition in Historical Documents **12**
3.1 Motivation 12
3.2 Dataset 13
3.3 Line-based OCR 14
3.3.1 Convolutional Recurrent Neural Network Classifier 14
3.4 Synthetic Data Training 15
3.5 Experiments 17
3.5.1 Real Dataset Experiment 17
3.5.2 Synthetic Dataset Experiment 18
3.5.3 Synthetic and Real Dataset Experiment 19
3.6 Conclusions 20

4 Text Categorization **21**
4.1 Semantic Space 21
4.2 Related Work 22
4.3 Multi-lingual Document Classification 23
4.3.1 Experiments 25
4.3.2 Conclusion 27

5 Dialogue Act Recognition **28**
5.1 Dialogue Act Recognition Task 28
5.2 Dialogue Act Recognition Related Work 29
5.3 Multi-lingual DA Recognition 30
5.3.1 Multi-lingual Model 30
5.3.2 Cross-lingual Model 30
5.3.3 CNNs Approach 31
5.3.4 Bi-LSTM Approach 31
5.3.5 Experiments 31

5.3.6 Conclusion 33

6 Aims of the Doctoral Thesis **35**

7 Conclusions **36**

Chapter 1

Introduction

Nowadays, the preferred research direction in machine learning is the usage of deep neural networks that learn parameters implicitly and do not require a manual feature engineering. The current trend is also to use advanced architectures, for instance encoder–decoder or seq2seq which obtained excellent results in machine translation or speech recognition. The main part of these architectures is often recurrent neural networks, which are covered in this report. There are successful applications of deep neural networks in social network analyses, sentiment analysis, named entity recognition and many others.

Furthermore, the importance of multi-lingual text processing increases significantly due to the extremely rapid growth of data available in several languages. Without multi-lingual systems it is not possible to acquire information across languages. One of the main goals of this report is to explore the multi-linguality in relevant tasks. The performance of many natural language processing (NLP) systems is strongly dependent on the size and quality of annotated resources. Unfortunately, there is a lack of annotated data for particular tasks and languages and manual annotation of new corpora is a very expensive and time consuming task. Moreover, the linguistic experts from the target domain are often required. The important contribution of this report is the presentation of the proposed neural network classifiers which can be used for a different set of tasks with minimal necessary changes.

The structure of this document is as follows. The first chapter provides a theoretical overview of popular state-of-the-art approaches in deep learning. The second one is focused on the OCR for handling historical documents in the German language. Following two chapters describe text categorization and dialogue act recognition. At the beginning of each chapter, there is an introduction and related work with the support of relevant publications. Moreover, the acquired knowledge is discussed and it is confronted against our proposed approaches. This report also presents our publications related to the specified topics. At the end of this report, the aims of the doctoral thesis and their analysis are outlined and the final chapter concludes the entire document.

Chapter 2

Deep Learning Survey

The current mainstream in machine learning is deep learning, more precisely deep neural network models. The essence of a deep learning approach is a deep multi-layer architecture which is even able to infer a feature vector from raw input data. So whether the input is a raw data or a high dimensional feature, the deep architecture can automatically extract high-level features necessary for classification or clustering [52].

In the traditional model of pattern recognition, a hand-crafted feature extractor gathers relevant information from the input and eliminates irrelevant variability. A trainable classifier then categorizes the resulting feature vectors into classes. In this scheme, we can use traditional approaches, including linear classifiers (e.g. logistic regression) or simple neural network models (such as multilayer perceptron).

A potentially more interesting scheme is to eliminate the feature extractor and feeding the network with raw inputs (e.g. normalized images). Then we can expect that backpropagation turns the first few layers into an appropriate feature extractor [32]. Considering, for example, bag-of-words vectors as a hand-crafted feature, the original text which has been turned to the bag-of-word vector is the raw input that has just been mentioned.

According to the survey in [52], if provided large enough data, well devised raw sequence based deep learning models can replace feature based deep learning models.

Basically, deep learning models fall into two groups. The first group consists of supervised approaches with deep architectures and the second group is unsupervised or semi–supervised generative models. In following sections, I introduce the most common models corresponding to the deep learning paradigm.

2.1 Deep Neural Networks

2.1.1 Recursive Neural Network

The main idea behind Recursive Neural Network (RvNN) is that a recursive structure can be found in various inputs (e.g. natural language sentences or scene images). The recursive structure can be used not only to identify the units but also to understand the interactions between them (see Figure 2.1) [48].

As we can see, the structure is an acyclic graph (i.e. tree). A vector representation is made for each tree level. Tree nodes can be considered as regions (i.e. part of an image, word, group of words etc.) The network is able to learn mappings from features (representa-

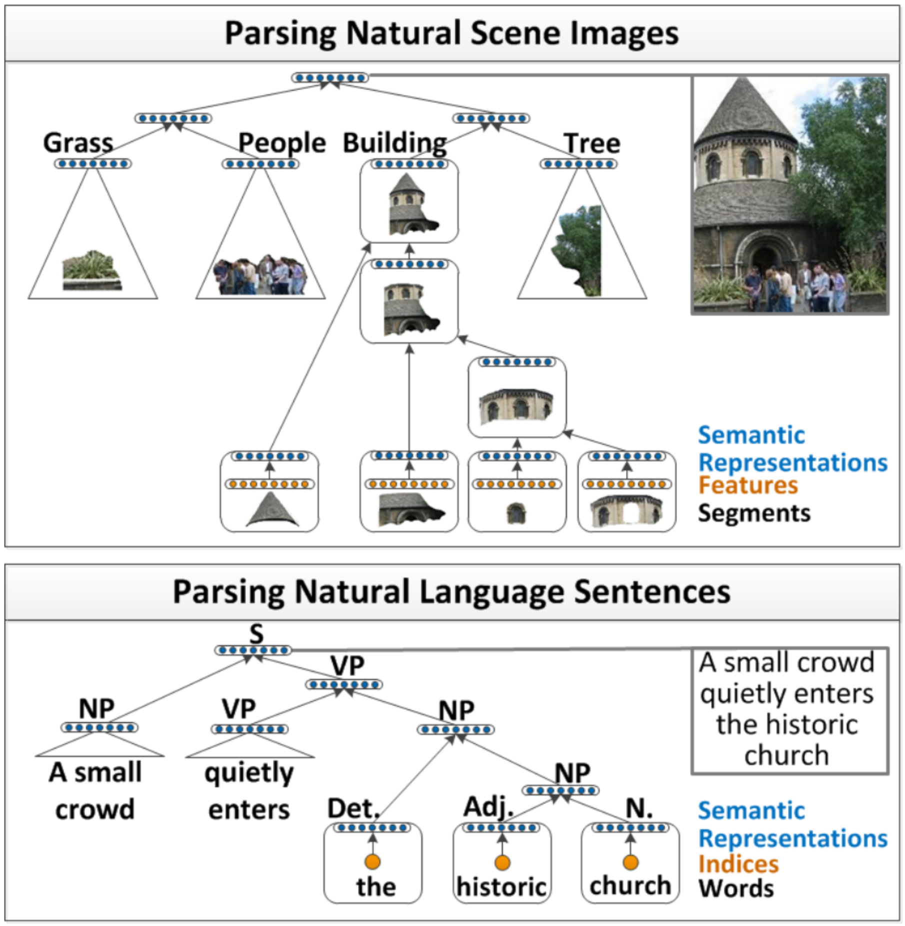


Figure 2.1: Illustration of the recursive neural network architecture [48]

tion of words or segments of interest in the image) to the high level semantic representation. The semantic representation space is then recursively merged to make the complete representation of an image or a sentence (see Figure 2.2).

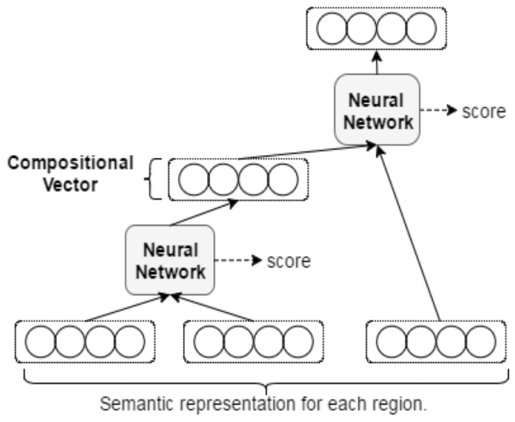


Figure 2.2: Example of merging [41]

Merging is done in the following way. RvNN calculates the score of a possible pair to merge them and build a syntactic tree. For each pair of units, RvNN computes a score for the plausibility of the merge. The pair with the highest score is then combined into a compositional vector. After each merge, RvNN will generate a larger region. The root of the RvNN tree structure is the compositional vector representation of the entire region (i.e. complete image or sentence) [48]. The backpropagation through structure (BTS) learning scheme was introduced to train the network [17], because it is capable to catch a tree-like structure.

2.1.2 Recurrent Neural Network

Recurrent Neural Networks (RNN) are currently preferred and widespread in NLP (see e.g. [2], [4], [7], [12] or [37]). In the previous section the tree-like structure can be found within a sentence. From a different point of view the sentence is a sequence of words and therefore each word has its context. This context is essential in order to understand the word. Figure 2.3 shows the illustration of the network which is related to the sequences.

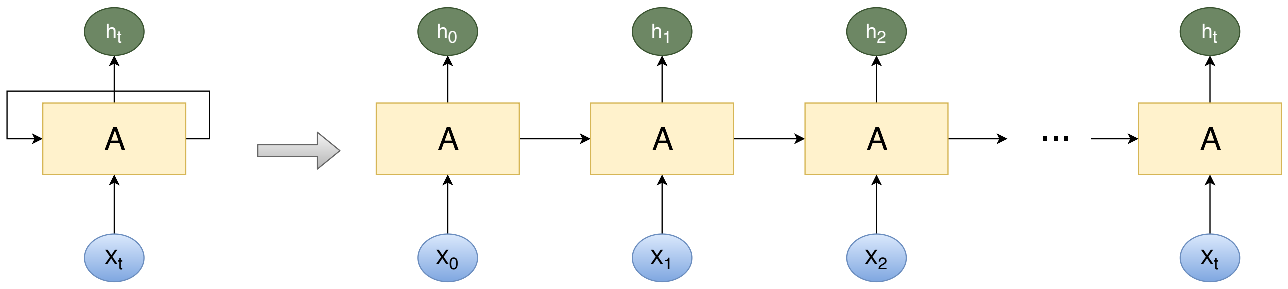


Figure 2.3: Illustration of the RNN

This kind of network (just like deep feedforward neural network) is prone to an unstable gradient problem. A study about understanding the difficulties of training a deep neural network has been provided by the authors of [16].

More precisely, the above-mentioned issue is either the exploding or the vanishing gradient problem. The gradient is calculated using the backpropagation algorithm and often the stochastic gradient descent is used to update weights in the network (i.e. for learning). The vanishing gradient problem involves weights in the earlier layers. If the weights in the earlier layers are about to update and the back-propagated gradient has a small value (i.e. vanishing value), the weights are barely updated. The weights in the earlier layers are mainly affected because the number of elements that influence the gradient calculation is large.

We have a very similar problem in the case of exploding gradient problem. Instead of a tiny updates of the weights in the earlier layers, too high updates are made. In brief, the gradients might decay or explode exponentially due to the multiplications of lots of small or big derivatives during the training [41].

Related to the Recurrent Neural Network, the network forgets the initial inputs with the entrance of the new ones and a decay of information through time occurs. One of the possible solutions to this problem is the usage of the Long Short-Term Memory [22] networks.

Long Short-Term Memory

In short, the Long Short-Term Memory (LSTM) is utilized to handle the above-mentioned gradient issue by providing memory blocks in its recurrent connections. Each memory block includes memory cells that store the network temporal states. Moreover, it includes gated units to control the information flow [41].

As we delve into the problem more deeply, we learn that the main drawback of the RNN is a long context. Considering a language model, if we try to predict the next word in a sequence by the closest history (e.g. the last few words), the RNN will learn the relation. It all depends on the distance or, more precisely, the size of the context. As that distance grows (i.e. long-term dependency), the RNN loses its ability to learn this relation. The LSTM is designed to avoid the long-term dependency problem. Furthermore, the long period information is a natural characteristics of this network.

Earlier I mentioned the gates (gated units). The gates are units which regulate the flow of information. Figure 2.4 shows the LSTM memory block and the interaction of gates and inputs.

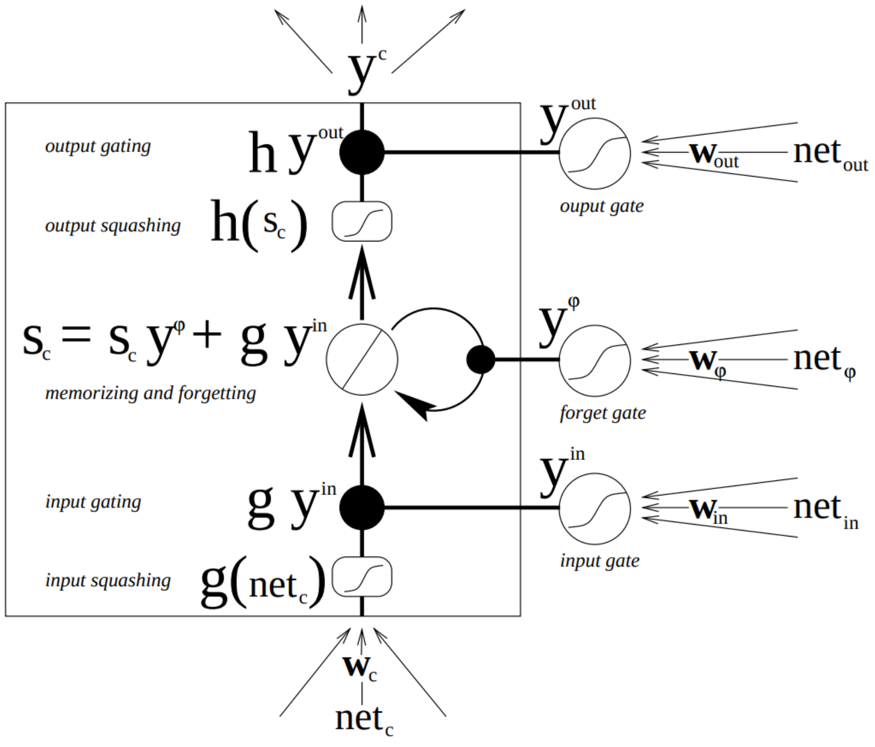


Figure 2.4: LSTM memory block with one cell [14]

The gates are composed of a sigmoid function and its output (values between zero and one) represents the regulation. There are three types of gates [14]:

1. *forget gate* which learns to reset memory cell contents once they are not needed any more,
2. *input gate* which decides which values should be updated,
3. *output gate* which decides which components of the output should be filtered out.

It is worth noting that in addition to the sigmoid function placed in the gates, the $tanh$ activation function is also present serving to determine candidate values to get added to the internal state.

There are several extensions and variants of the LSTM. The most popular one is the Gated Recurrent Unit (GRU) [9] which will be described int the following section.

Gated Recurrent Unit – GRU

The GRU model is simpler than a standard LSTM. It combines the forget and input gates into a single update gate. Since GRU is simpler than LSTM, it has fewer necessary operations and therefore the GRU trains faster than the LSTM.

In the GRU there is no such thing as a memory cell. The activation of gates in GRU only depends on current input and previous output. Figure 2.5 shows the comparison of the LSTM and the GRU.

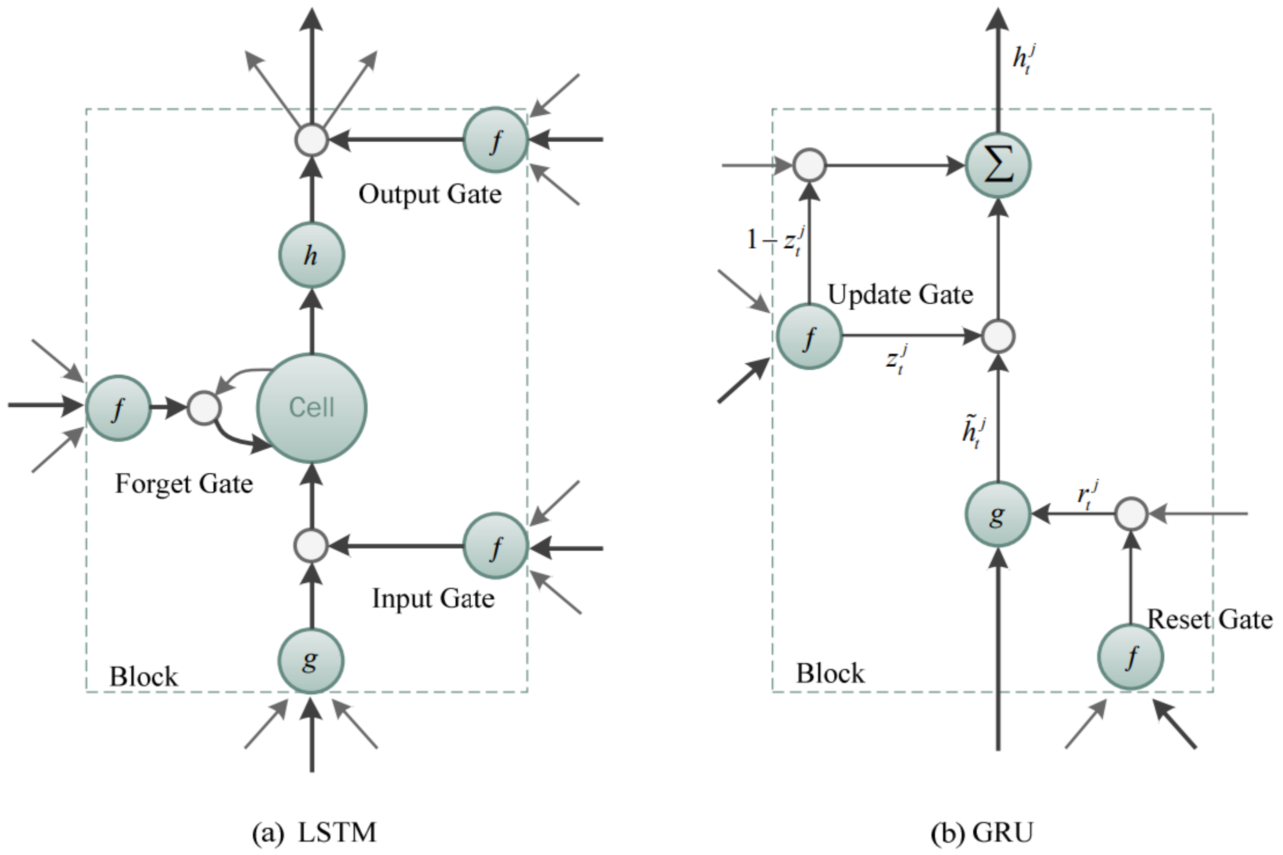


Figure 2.5: Comparison of the structures of the LSTM and the GRU [51]

In general, one cannot say which one is better in terms of application, it strongly depends on the domain and the input characteristics. From my point of view, however, the use of LSTM is slightly predominant.

Seq2Seq Architecture

Another important chapter in the RNN field is the seq2seq architecture. An inherent part of the RNN is a capability to process sequences. The seq2seq model is basically a function which maps the input sequence to the output sequence. An encoder-decoder architecture is often used as the function [8].

In a nutshell, the encoder translates and compresses the input sequence into a fixed length context vector. This vector should represent a meaning (semantic information) of the whole input sequence. The context vector is an input to a decoder. The task of the decoder is to process the context vector and thereafter generate the output sequence.

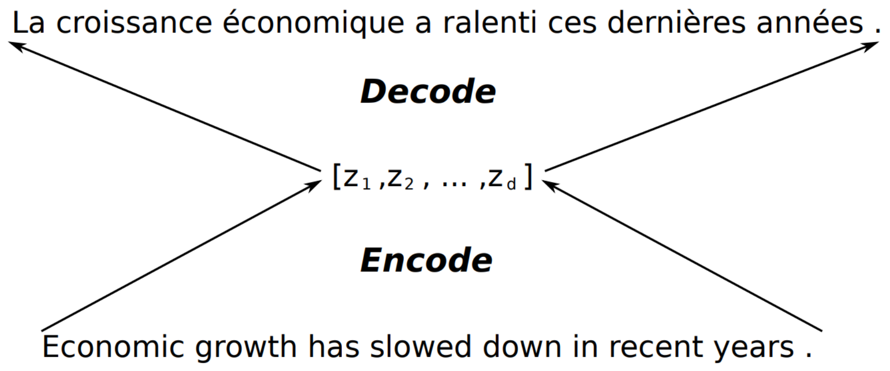

La croissance économique a ralenti ces dernières années .
Decode
[z1 ,z2 , ... ,zd]
Encode
Economic growth has slowed down in recent years .

Figure 2.6: Encoder-Decoder architecture [8]

This kind of architecture was originally developed for machine translation (see Sutskever et al. [50] or Cho et al. [8]). Both the encoder and decoder are recurrent neural networks (most often an LSTM or a GRU). The main drawback of this architecture is fixed-length of the context vector. In the case of long sequences, it is very hard to remember the earlier elements. In order to handle this issue, the attention mechanism has been developed.

Attention Mechanism

Bahdanau et al. [2] consider a fixed-length vector in the encoder-decoder architecture as a bottleneck in improving the performance of this basic architecture and propose an approach when the decoder decides parts of the source sentence to pay attention to.

Figure 2.7 shows an example of the machine translation between English and French. Points of intersection show the correlation between the source and target words. We can also appreciate the attention mechanism when generating "European Economic Area", because in French, the order of these words is reversed "européenne économique zone". Such a variety in the order of words in different languages is a matter that is well addressed by the attention. In the Bahdanau's implementation, each pixel shows weight α_{ij} of the annotation of the j-th source word for the i-th target word in grayscale.

The interpretation of the attention in deep learning is mostly as a vector of importance weights. It tries to estimate how strongly is the predicted element correlated with the other elements and it takes the sum of their values weighted by the attention vector as the approximation of the target.

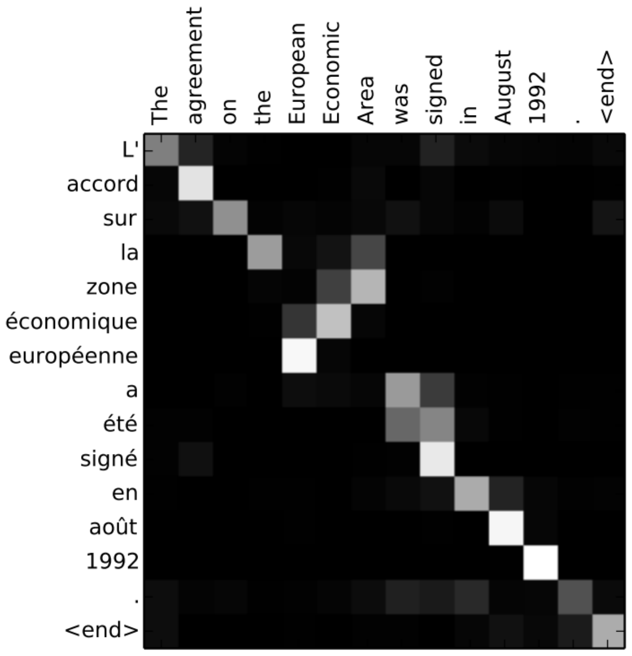


Figure 2.7: Example of the attention-based machine translation

From the point of view of the previous section, while the context vector has access to the entire input sequence, the forgetting of the earlier elements is not an issue anymore.

There are also other uses of the attention mechanism. Xu et al. [54], inspired by recent work in machine translation and object detection, introduce an attention based model that automatically learns to describe the content of images (see Figure 2.8).

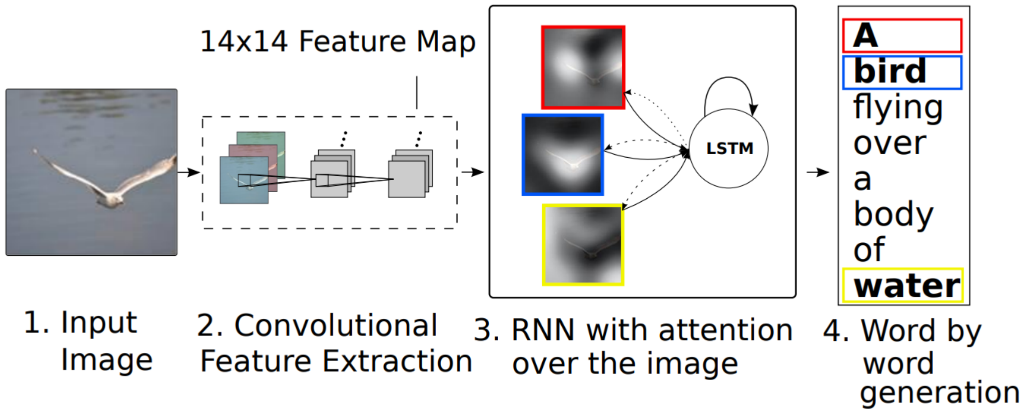

14x14 Feature Map
LSTM
A
bird
flying
over
a
body
of
water
1. Input Image
2. Convolutional Feature Extraction
3. RNN with attention over the image
4. Word by word generation

Figure 2.8: Model for image description with attention [54]

Another example of the attention present Cheng et al. [7]. Figure 2.9 shows the illustration of the proposed model while reading the sentence. The red color represents the current word, the blue represents memories. Shading indicates the degree of memory activation.

Vaswani et al. [53] go even further with a machine translation task and propose a simple network architecture, the Transformer, based solely on attention mechanisms with avoidance

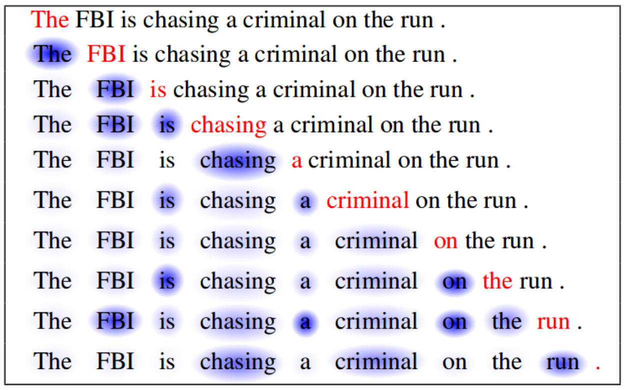

The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .

Figure 2.9: Illustration of the processing the sentence with attention [7]

of the recurrent or convolutional layers. It is worth noting that for translation tasks, the Transformer can be trained significantly faster than standard seq2seq architectures.

2.1.3 Convolutional Neural Network

The convolutional neural network (CNN) was originally developed for image recognition and it is one of the most noticeable representatives of deep learning methods.

Convolutional networks combine three architectural ideas to ensure some degree of shift and distortion invariance [32]:

1. local receptive fields,
2. shared weights (or weight replication),
3. spatial or temporal subsampling.

Thanks to these properties, they are extremely suitable for processing images and other multidimensional inputs, because they allow us to use a raw input (e.g. image), without necessity to create hand-crafted and sometimes cumbersome features. Figure 2.10 shows an example of CNN architecture.

Typically in CNN, there is a number of convolutional layers followed by pooling (subsampling) layers. As a final layer, it is usually used a fully connected layer.

In each convolutional layer, there are several filters (kernels) k of size $n \times m \times q$[1]. The filters are the base of local connections that are convoluted with the input and share the same parameters to generate feature maps. The convolutional layer computes a dot product between the weights and its inputs. Thereafter, in the subsampling layers, each feature map is downsampled to decrease the number of parameters. Thanks to the subsampling, the natural property of the CNN is the reduction of dimensionality because with each increasing layer with fewer neurons, the number of parameters decreases and the dimension as well. In the final consequence, it reduces the risk of overfitting.

Thanks to the same shared weights (e.g. kernel 4×4, requires only 16 learnable weights), the CNN does not have a problem with the above-mentioned vanishing or exploding gradients problem (regardless of an image size).

[1]If we consider an image as a input, the parameters refer to the width, height and color depth of an image

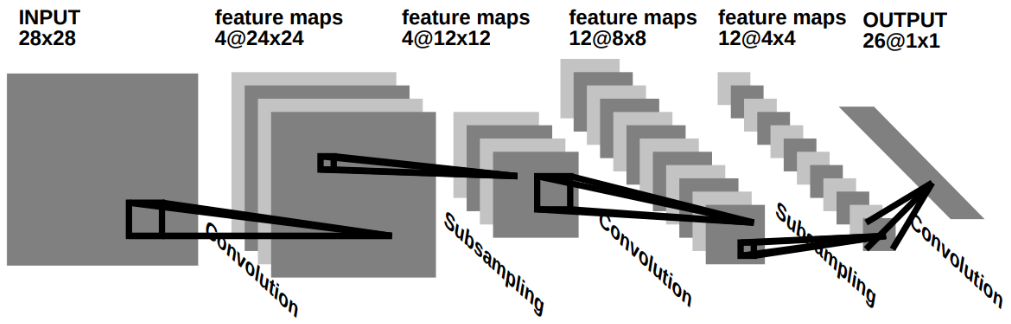


Figure 2.10: Example of the CNN for image processing [32]

Subsampling is a way to reduce the resolution of the feature maps, resulting in a fewer number of parameters and a dimension reduction.

Subsampling – Pooling

Figure 2.11 shows two main subsampling methods.

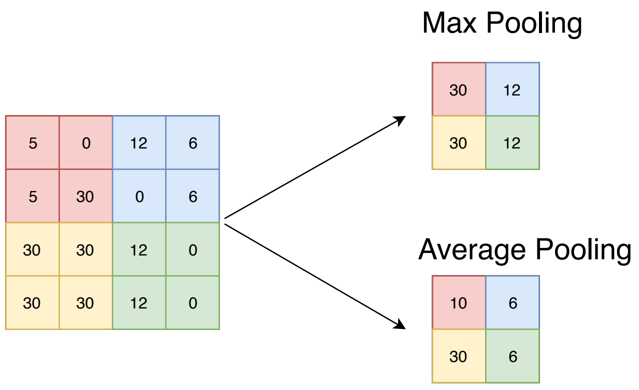


Figure 2.11: Illustration of the max and average pooling

In general, max pooling extracts the most important features like edges. Average pooling takes basically all, even features which might not be important for object detection.

Although CNN is probably the most popular method in computer vision tasks, attempts to utilize it in the NLP has been made. Zhao and Wu [55] propose a novel convolutional neural network with the attention mechanism to improve the performance of sentence classification. They confirm the presumption in the previous section that in traditional deep learning method it is not easy to encode long term contextual information and correlation between non-consecutive words effectively. In contrast, their attention-based CNN is able to capture these kinds of information for each word.

2.2 Summary

Summing up, I have presented the most important deep learning neural networks and their issues, such as difficulty to train or suffering from vanishing and exploding gradient problem. In the case of the RNN, I presented concrete examples which are popular and fight with the issues (i.e. the LSTM and the GRU) and I also briefly introduced a seq2seq, encoder-decoder architecture and the attention mechanism.

Moreover, I shortly presented a CNN with a quick overview of Max and Average Pooling and I showed a relation to the NLP and to the attention mechanism.

Chapter 3

Optical Character Recognition in Historical Documents

This chapter is focused on applications of neural networks for optical character recognition (OCR) in Historical German documents. By researching this area, I have spent the most of my work in postgraduate study so far. Within this chapter, I first introduce the task and its motivation and then I describe the models used for recognition. Finally, I conclude this chapter, by providing experiments and future work.

3.1 Motivation

The efforts to preserve the cultural heritage of historical documents has become a significant task in the document processing field. A reasonable information retrieval on scanned documents is possible only after the text recognition process though. Optical character recognition and handwritten text recognition (HTR) need to be implemented with a respect to the historical domain. In such a domain, we often struggle with the quality of historical document and with a lack of annotated data.

Before the recognition process, we need to segment page to locate text blocks. Then, it is necessary to provide text lines from the extracted text blocks. In the next phase, it is possible to segment a text line into images of individual characters.

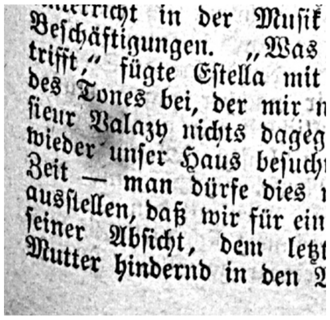
If we consider character classification, the classifier must learn the visual models of individual glyphs, which requires multiple instances of each character sample. A manual annotation is typically a very time consuming action. On the other hand, synthetic data (i.e. automatically created samples) are easy to obtain. A combination of synthetic and manually annotated data is used in this part of research. In the following sections, I introduce proposed approaches and experiments.

3.2 Dataset

As I mentioned earlier, our dataset consists of German historical documents. Even though a German text in the form of a scanned document is printed, recognition becomes difficult for several reasons:

- german text is rendered by the historical (Fraktur) font,
- documents often have a non-uniform layout (e.g. 3.2),
- documents have usually poor image quality (i.e. waved or warped text, text skew, noise, etc.).

Figure 3.1 shows an example of the waved text. It is usually present at the inner edges of the documents.



terricht in der Muſik
Beſchäftigungen. „Was
trifft," fügte Eſtella mit
des Tones bei, der mir n
ſieur Balazy nichts dageg
wieder unſer Haus beſuch
Zeit — man dürfe dies
ausſtellen, daß wir für ein
ſeiner Abſicht, dem letzt
Mutter hindernd in den 2

Figure 3.1: Example of the waved text

Our dataset consists of 10 annotated pages of German historical newspaper (e.g. Figure 3.2).

Figure 3.2: Example of a historical newspaper

3.3 Line-based OCR

Line-based OCR approaches are currently very widespread and utilized by state-of-the-art systems. They use a whole text line as an input instead of individual characters. Therefore, it is not necessary to perform character segmentation, which is often error-prone, due to the poor quality and fragmented characters.

A combination of deep convolutional and recurrent neural networks (CRNN) is nowadays used for line-based OCR [4]. An LSTM is often used within the CRNN model. For this task, we used the CRNN (see for example [20] or [47]) model inspired by Shi et al. in [43].

To be able to train such a line-based OCR system, we need to provide a big annotated corpus. Manual annotation is a very time-consuming way of dataset creation and hence our annotated corpus consists of only ten manually annotated pages (1368 images of text lines). We consider the training set to be too small and inappropriate for training a neural network classifier.

The lack of training data is a significant issue with regard to the ability to learn a language model. Since it was shown that an implicit language model is an important part of the line-based OCR system [42], we have here an important observation and relation to the NLP field.

3.3.1 Convolutional Recurrent Neural Network Classifier

Proposed neural network classifier is a combination of a convolutional neural network and a recurrent neural network and shares similarities with the work of Shi et al. [43]. We use the Connectionist Temporal Classification (CTC) loss as proposed by Graves et al. [18].

Binarized images are used as input from which the CNN extracts features. These are passed as an input to the RNN, which uses LSTM cells. Following Graves et al. [19], we use a bidirectional LSTM architecture.

The output from the bidirectional LSTM layer is a set of dense layers with softmax activation function, representing a probability distribution of characters per time frame. Let $\mathcal{A}$ be a set of symbols that the classifier recognizes ($|\mathcal{A}| = 83$). Then $p_t^{a_i}$ is a probability of observing character a_i at given time t. At each time t the sum of the probabilities of all symbols is equal to 1.

$$\sum_{i=1}^{|\mathcal{A}|} p_t^{a_i} = 1 \tag{3.1}$$

The most probable symbol $\hat{a}_t$ of each time frame t is then determined as:

$$\hat{a}_t = \operatorname*{argmax}_{a_i \in \mathcal{A}} p_t^{a_i} \tag{3.2}$$

The last part of the classifier is a transcription layer. It decodes the predictions for each frame into an output sequence. To be able to distinguish each individual characters the blank-symbol (-) is inserted. It is also necessary to remove any duplicates in the sequence. The architecture of the classifier is depicted in Figure 3.3.

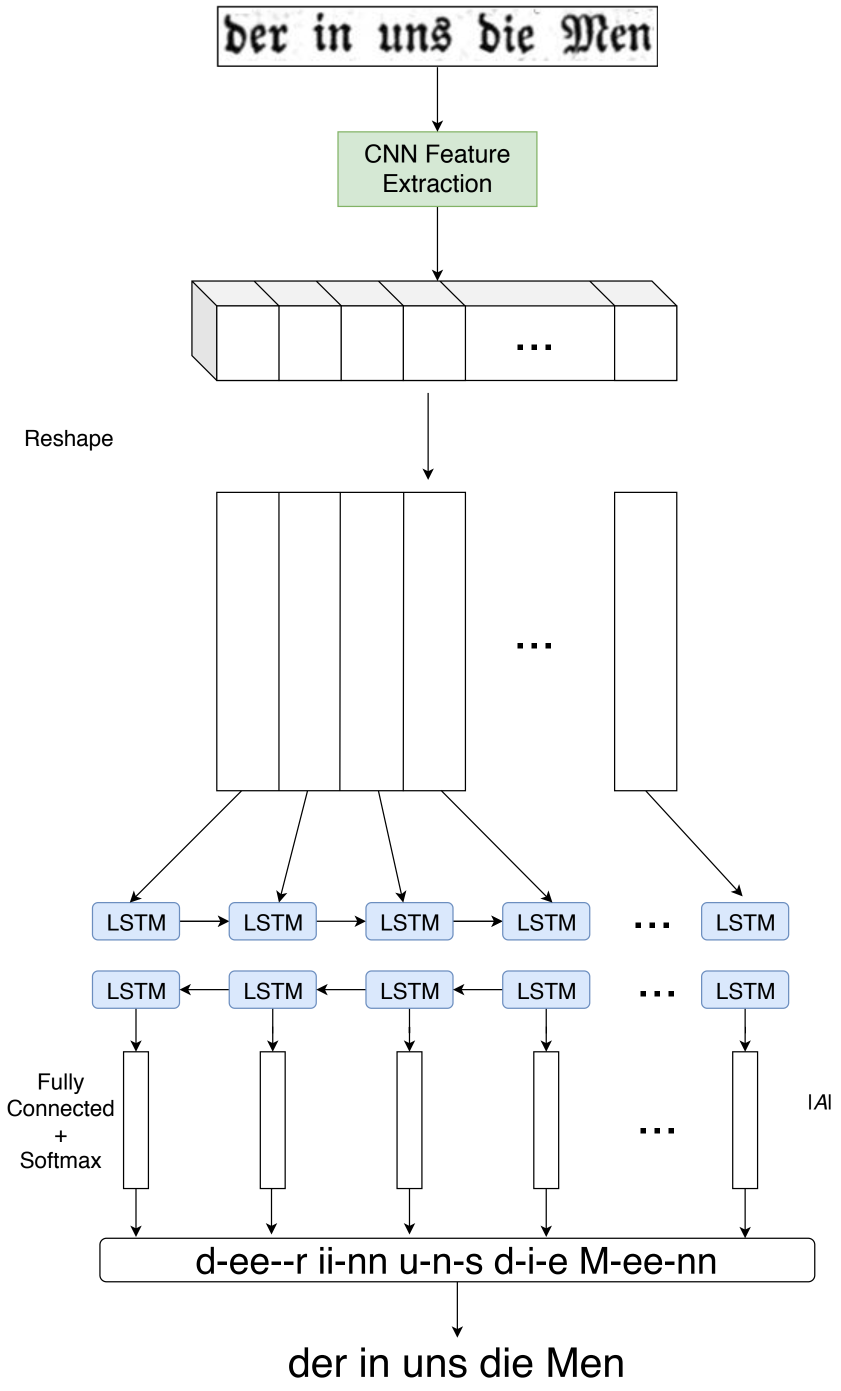


Figure 3.3: CRNN architecture

3.4 Synthetic Data Training

Earlier, I mentioned that in historical domain we often struggle with a lack of training data (in our case only 10 annotated pages). Therefore, a following training data creation method has been proposed. The idea is to train the classifier in two steps. First, the network is trained on a large amount of synthetic data (i.e. artificially created text lines). The synthetic data should be sufficient to initialize the statistical language model which plays an important role in line-based classification. In [42] authors have shown that the implicitly learned language model can improve the recognition accuracy. Therefore, the texts used for synthetic data generation should be aimed at the domain, we work in[1]. The synthetic data should also provide a basic glyph model. For the second stage, we use a relatively small dataset of real text lines for additional training, which will fine-tune the classifier.

Using synthetic data for OCR is a considerable step in historical documents processing. Jaderberg et al. present a framework for recognition and synthetic data generation [24]. Margner et al. [33] present synthetic data for Arabic OCR and another synthetic data generation was proposed by Gaur et al. [13].

There is a number of tools (i.e. tool for Arabic OCR in [33] or Aletheia proposed in [10]) that deal with the synthetic data generation and annotation. One of our synthetic datasets utilizes the Ocropus OCR System (more precisely OCRopy-linegen) [3], which is another example of a tool for text recognition and synthetic data generation. There are more ways to create synthetic text lines. I present two possible methods in the rest of this section.

Semi-automatic Synthetic Data Creation

First of all, we must provide a text source in the target domain. Then, the simplest way to create an image of the text line is to take images of individual characters and put them together. It must be done with a respect to the target font though.

It is very convenient to have several different examples of each symbol. Then we ought to use a randomly picked image for a given symbol to reach a greater diversity of the synthetic dataset. This way can be considered as a simple form of data augmentation technique (e.g. Perez et al. [40]) because from one source text line it is possible to create several different corresponding images, just by the random choice of the character image.

Another thing to consider is gaps between characters. The gap should be chosen with regard to a font style. The simplest solution is to estimate a constant pixel value and use it as a gap for each pair of letters. Another way is to specify a range (e.g. 1 px – 5 px) and use a random value within this range. Finally, it is possible to use real examples to provide precise values of gaps between each pair of characters. However, because of the quality of scans and noise, the extracted precise values could be distorted. Therefore we chose to use the random gaps approach with a range 1 px – 5 px. We refer this way of synthetic data creation as the random space method. Figure 3.4 shows an example of generated random space text lines.

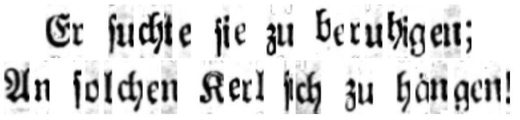
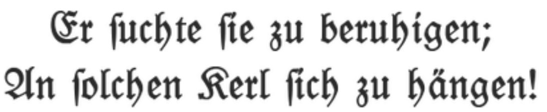

Er ſuchte ſie zu beruhigen;
An ſolchen Kerl ſich zu hängen!

Figure 3.4: Examples of generated text lines with random gaps between characters

[1]In our case historical German texts from the second half of the nineteenth century.

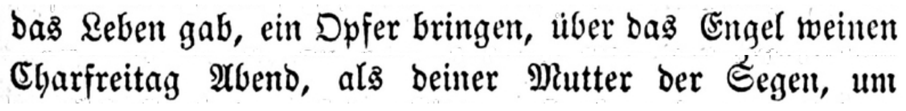
Synthetic Data Generator

An alternative way of creating such a synthetic dataset is to use a text generating tool (e.g. OCRopy-linegen [3]). The tool usually has the possibility to choose a font, source texts and several types of image transformations (skewing, warping etc.) to make the desired image of a text line. Every rendered character is always the same and then it could be very useful in the case of printed text (see Figure 3.5). The images of annotated text lines are depicted in Figure 3.6.



Er ſuchte ſie zu beruhigen;
An ſolchen Kerl ſich zu hängen!

Figure 3.5: Examples of generated data by OCRopy–linegen



das Leben gab, ein Opfer bringen, über das Engel weinen
Charfreitag Abend, als deiner Mutter der Segen, um

Figure 3.6: Examples of real dataset lines

3.5 Experiments

In this section, I present experiments which verify and evaluate the proposed approach. The original real images have an 8-bit grayscale png format. The images were binarized using the Otsu's method [38] from OpenCV[2] library. After the first set of experiments, we discovered that binarized images obtained better results than grayscale ones, which is in compliance with Gupta et al. [21]. Hence, the binarized images are used for all experiments. Since the dataset combines the real annotated images and the synthetic ones, there are more ways to train a classifier. In the following sections, three different experiments of training setups are presented and evaluated.

3.5.1 Real Dataset Experiment

This experiment employs only the real annotated images, completely without the synthetic ones. Seven pages of the annotated dataset is used to train a CRNN. The results of this baseline approach, as well as the course of training, are depicted Figure 3.7.

The left image shows train and validation CTC loss, while the right one shows the average character error rate (CER) and the average word error rate (WER). The values are averaged through all validation line images. The average edit distance is a number of operations (removal, insertion, substitution) required to transform one predicted text line into the correct one.

[2]https://opencv.org/

Either of these images shows that after approximately 80 epochs of training the model reaches the best results and then starts to stagnate. Table 3.1 shows the results after 100 epochs.

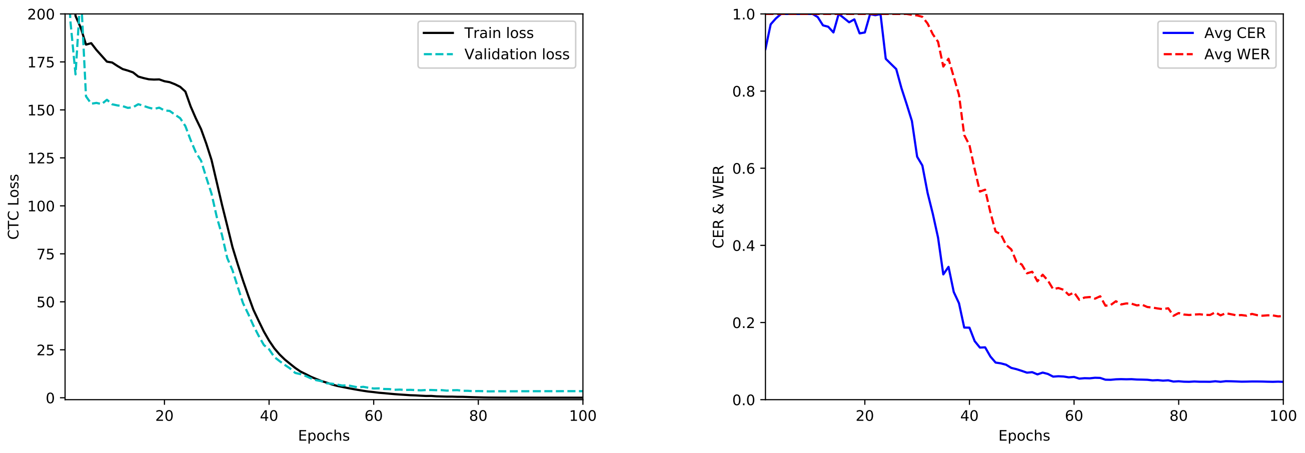


Figure 3.7: WER, CER, Train & Val loss after 100 epochs of the training annotated real dataset

Train loss	Val loss	WER	CER	Edit distance
0.056	3.030	0.068	0.014	0.464

Table 3.1: Results of the real dataset experiment after 100 epochs of training

The experiment showed that the model reached excellent results on the validation data and no significant overfitting occurred. On the other hand, the main drawback is that limited implicit language model has been learned (only 955 unique text lines). Another important issue is a very long learning time (at least 80 epochs).

3.5.2 Synthetic Dataset Experiment

This experiment compares OCR results on the synthetic dataset after 25 epochs of training (see Figures 3.8 and 3.9). It verifies both of the above-mentioned methods for synthetic data generation.

Figures show that the overfitting occurs very quickly in both cases (see validation loss). The curves also indicate that training the model longer than for 5 or 10 epochs is not beneficial. Table 3.2 shows the results. We can conclude that the Random space dataset is more similar to the real data, because the validation results are better than in the case of OCRopy-linegen.

	Train loss	WER	CER	Edit distance
Random space	7.630	0.655	0.237	10.732
OCRopy-linegen	0.014	0.734	0.281	12.174

Table 3.2: The results of the random space and OCRopy-linegen dataset

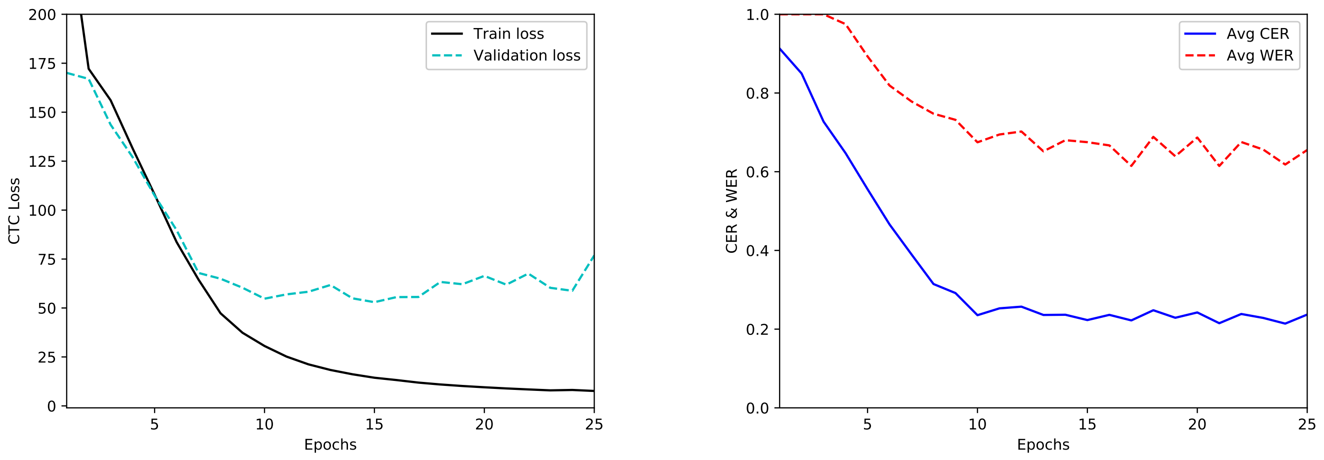


Figure 3.8: WER, CER, Train & Val loss after 25 epochs of the synthetic random space dataset method.

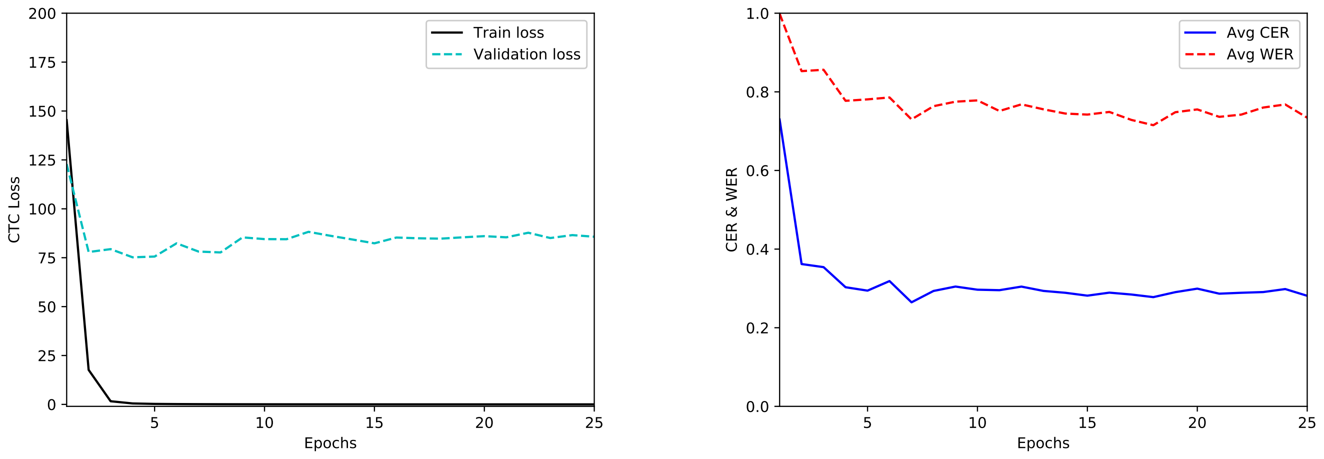


Figure 3.9: WER, CER, Train & Val loss after 25 epochs of the dataset from the synthetic data generator (OCRopy-linegen).

3.5.3 Synthetic and Real Dataset Experiment

The last presented experiment is the CRNN training with a combined dataset (synthetic and real line images). The synthetic dataset was used to initialize the classifier with 10 epochs of training. Then, the annotated training dataset was used to perform another 25 epochs of additional training (see the first part of Table 3.3). The middle part of the table shows the results from the synthetic dataset only. The last line of the table shows the results completely without the synthetic dataset. The combination of the synthetic and the annotated real dataset obtained the best results.

	Train loss	**WER**	**CER**	**Edit distance**
Synthetic and real dataset experiment				
Random space dataset init	0.069	**0.065**	**0.012**	**0.428**
OCRopy–linegen dataset init	4.838	0.307	0.069	2.725
Synthetic dataset experiment				
Random space	7.630	0.655	0.237	10.732
OCRopy-linegen	0.014	0.734	0.281	12.174
Real dataset experiment – 100 epoch	**0.056**	0.068	0.014	0.464

Table 3.3: Summary of the results for all experiments

3.6 Conclusions

From my point of view, since the character segmentation is error-prone, the line-based approach is more promising. The main drawback of the line-based approach is the size of a training dataset. Hence the experiments with a synthetic dataset have been made which achieved excellent results. The future work will be focused on the OCR error correction method. We will use the statistical language model and the seq2seq method to transform the predicted result to the most probable correct text line.

Chapter 4

Text Categorization

Text categorization (or text classification) is an important discipline in NLP (spam filtering, sentence classification, clustering and so on). Automatic text categorization is also one of the main components in searching systems for providing additional related results.

The following section is devoted to semantic spaces since it is an important part of NLP tasks. Thereafter, I briefly provide related work and a quick overview of the state-of-the-art approaches. I also present our experiments and concrete application of deep neural network in a text categorization task.

4.1 Semantic Space

Since the essential part of the following proposed approaches is a word vector representation, I provide a brief introduction to the semantic space which is directly related to the word vector representation. According to Brychcin [5], a semantic space S is a function which projects word w from vocabulary **V** into Euclidean space with dimension d ($S : \mathbf{V} \mapsto \mathbb{R}^d$). The meaning of the word w is represented as a real-valued vector $S(w)$. These real-valued vectors are representations of the words, so-called word embeddings. They should be capable of capturing the meaning (semantic information) of words. So the semantically close words should be located, in the terms of Euclidean or Cosine distance, close to each other within the space. Figure 4.1 shows the example of projected word vectors. For the word *Berlin* we can see other German cities very close in the space. There are also some other capital cities, predominantly in Europe. Even some German words can be found.

Word vectors can be computed by word2vec [37], glove [39] or fastText [27] models. The advantage of the fastText is possibility to determine a vector to a word beyond a vocabulary based on n-grams. In general, word vectors can be created and trained using a deep neural network with an embedding layer. It is also possible to fine-tune the pre-trained word vectors to the target domain.

Unfortunately, it is very hard to create a representation of the words throughout more languages. Two same sets of words in two different languages might have different semantic spaces due to the different language characteristics. In order to address the multi-linguality, the unification of the spaces is appropriate[1]. One of possible ways to unify the semantic spaces is to use a linear transformation.

[1]This unification can be omitted, for example, by machine translation.

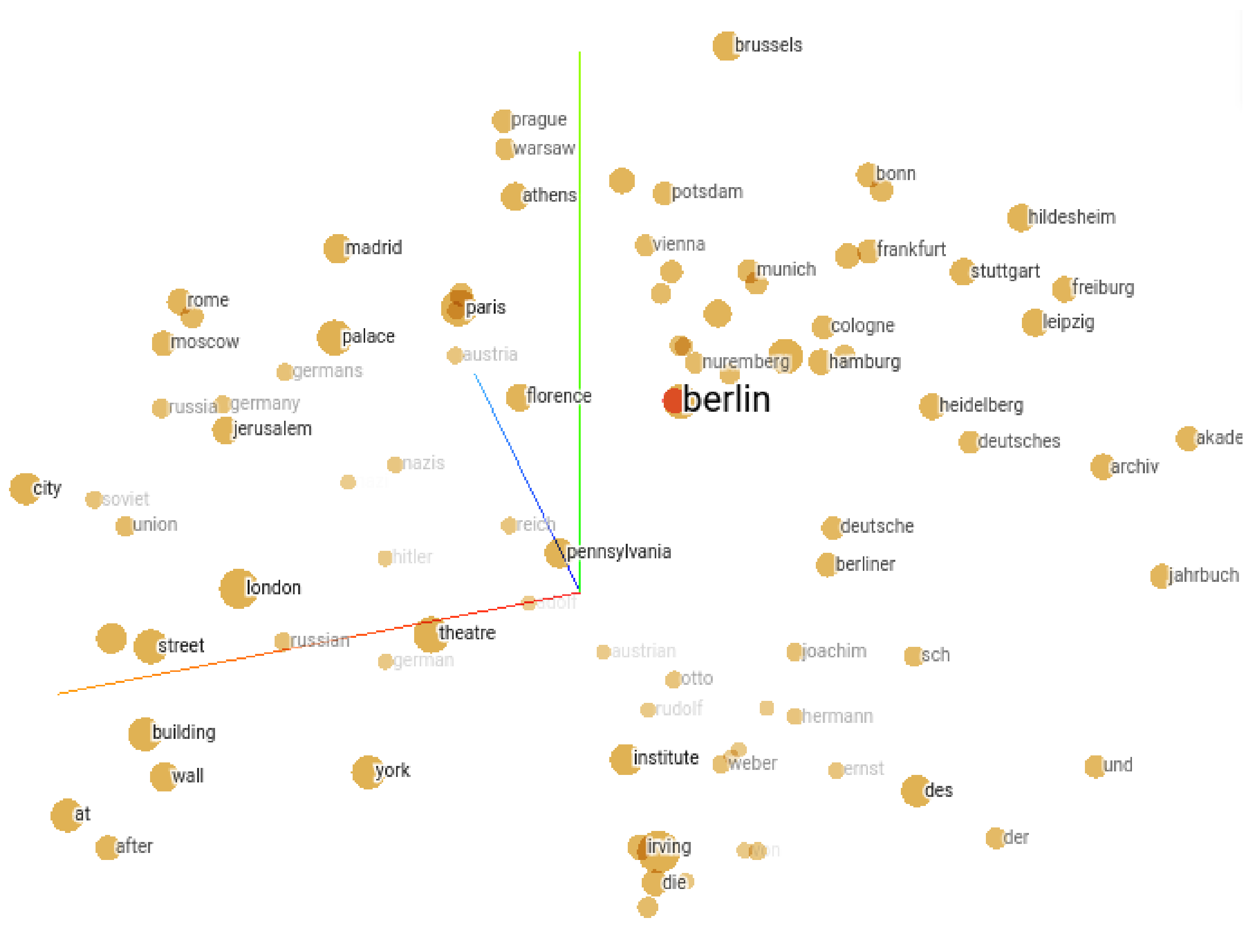


Figure 4.1: Example of projected word2vec word vectors (https://projector.tensorflow.org/)

In a nutshell, linear transformation can be expressed by a transformation matrix **T**. By the multiplication of the vectors with this matrix, the scaling, translation or rotation can be performed, which results in a different space. It is worth noting that Brychcin in [5] explores unsupervised techniques to achieve the unified semantic space for different languages.

4.2 Related Work

The current state-of-the-art approaches for text categorization tasks are based on deep learning. The most often recurrent neural networks are used (for example [26], [23]). In Section 2.1.3 Convolutional Neural Network, I indicated the usage of CNN in the NLP. The input of the CNN is a tensor. Regardless of whether the tensor represents an image or just a weight matrix (e.g. tf-idf values or word vectors), CNN is able to extract features anyway.

Kim trained a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model (i.e. word2vec). It uses three sizes of convolutional kernels - $(3, E)$, $(4, E)$ and $(5, E)$ where E is the embedding dimensionality. 100 kernels of each size are computed simultaneously and their outputs are merged and fed into a fully connected layer. The architecture is depicted in Figure 4.2.

As we can see, the shape of the convolutional kernel is designed to catch both dimensions (i.e. time dimension as well as the embedding dimension). Kim made set of experiments in

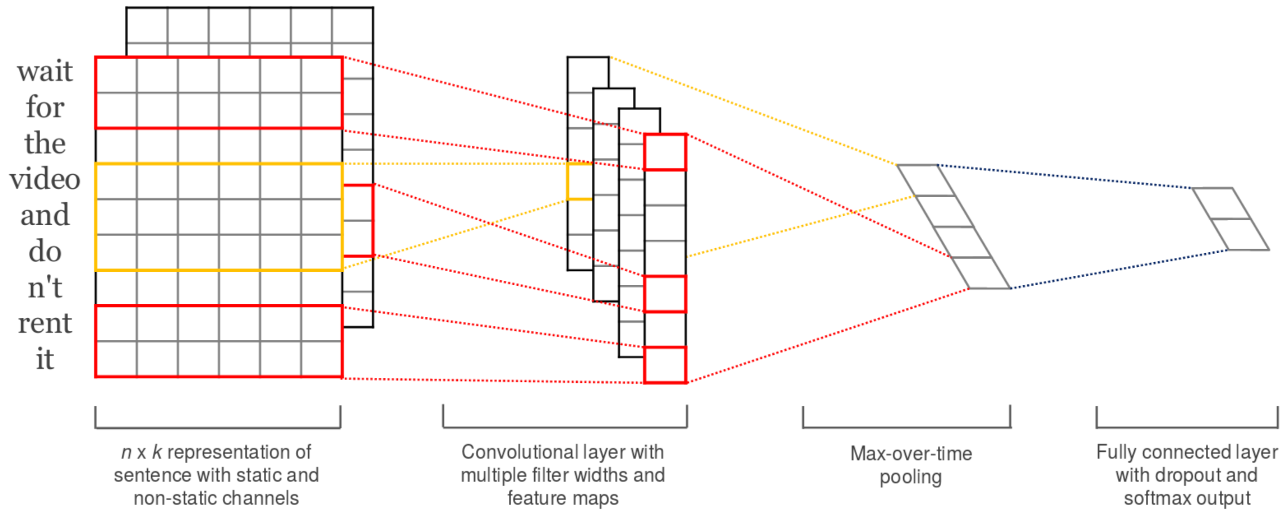

wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 4.2: Architecture of the KIM approach

the area of sentence classification. He used a pre-trained word embeddings model (made by word2vec [37]) as an embedding layer and he compared it with a random initialized vectors. Broadly speaking, in all but a few cases it has been proved that the usage of pretrained word embeddings is beneficial when they are allowed to train and even if they are kept static they are better than trained random-initialized ones. Despite little tuning of hyperparameters, this simple model achieves excellent results [29].

From other deep learning methods, we can mention Conneau et al. [11]. They try to learn a high-level hierarchical representation of a sentence and at the same time operate with a low level representation of the texts – the characters. Contrary to the Kim's approach, the proposed architecture is much deeper.

An LSTM approach is presented by Johnson and Zhang in [26]. They use a sophisticated region embedding and their results indicate that embeddings of text regions, which can convey complex concepts, are more useful than embeddings of the isolated words. In general text categorization task, there are successful applications of Support Vector Machines (e.g. benchmark made by Lewis et al. [15]). Typical hand-crafted features for the linear classifier (such as SVM) include bag-of-words (with a usage of tf-idf) or n–gram model. At present, the non-linear methods (e.g. deep neural networks) are preferred though.

4.3 Multi-lingual Document Classification

In this section, I present our proposed classifier for multi-lingual and multi-label[2] document classification. It has been inspired by the Kim's architecture which I mentioned above.

The convolutional network we use for classification requires that the inputs have the same dimensions. Therefore, the documents with fewer words than a specified limit are padded, while the longer ones must be shortened. This is different from Kim's approach where documents are padded to the length of the longest document in the training set. We are working with much longer documents where the lengths vary significantly. Therefore, the shortening of some documents and thus losing some information is inevitable in our case. The

[2]Document can be labeled with one or more classes.

architecture of the network is depicted in Figure 4.3. The figure shows the processing of two documents in different languages (English and German) by our network. Each document is handled in one training step. The key concept is the shared vocabulary and the corresponding shared embedding layer.

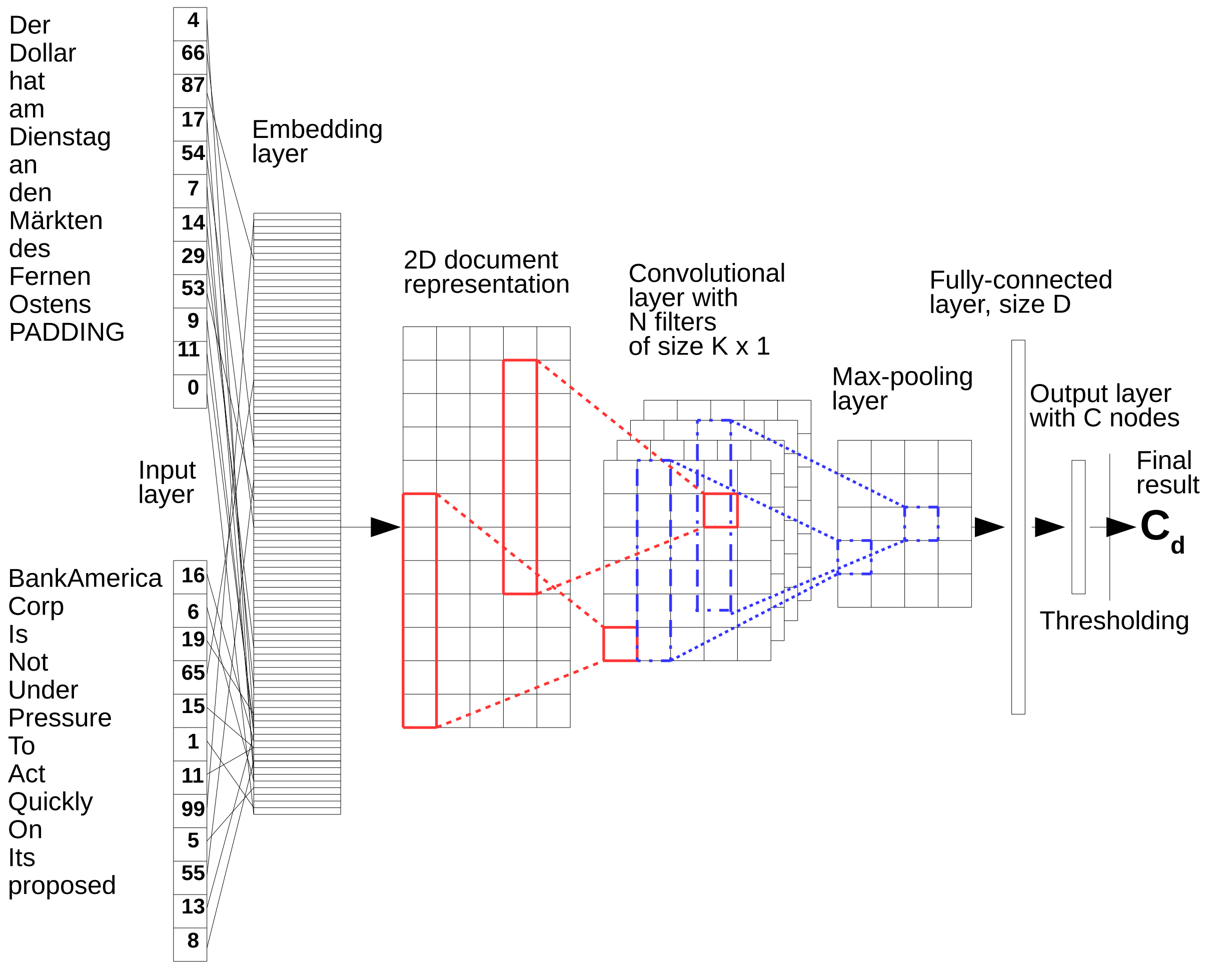


Figure 4.3: The classifier architecture

The input of our network is a vector of vocabulary word indexes of the length M where M = 100 is the number of words used for document representation. The second layer is an embedding layer which represents a look-up table for the word vectors. It translates the word indexes into word vectors of length E = 300. The document is then represented as a matrix with M rows and E columns. The next layer is the convolutional one. We use N_C = 40 convolution kernels of the size $K \times 1$ ($K = 16$) which means we do 1D convolution over one position in the embedding vector over K input words. The following layer performs a max-pooling over the length M - K + 1 resulting in 40 $1 \times E$ vectors.

The output of this layer is then flattened and connected to a fully–connected layer with E nodes. The output layer contains $|C|$ nodes where $|C|$ represents the number of categories we consider. The output of the network is then thresholded to get the final results. The values greater than a given threshold indicate the labels that are assigned to the classified document. In the case of multi-label classification, we utilize the sigmoid activation function in the output layer.

The proposed architecture allows us to use two training approaches. The first one utilizes the shared vocabulary for all available languages. In such case, it is not necessary to perform any translation or linear transformations of the semantic space. The following experiments show the results with this setup.

In the second approach, the classifier is trained by the word vectors of one language only. All other semantic spaces need to be transformed into the one used for training. Our overview of semantic space transformations in document classification can be found in [35].

4.3.1 Experiments

For the experiments we employed the Reuters RCV1 and RCV2 dataset [1]. RCV1 contains a large number of English documents, while the RCV2 is a multi-lingual corpus that contains news stories in 13 languages. We focused on four languages, namely English, German, Spanish and Italian. We used 20,000 most frequent words from each language for creation the shared vocabulary.

Single-label Classification

We present three possible settings of the embedding layer. The first one uses static word2vec embeddings (Word emb notrain), the second one uses word2vec embeddings which are fine-tuned during the network training (Word emb train) and the last one uses randomly initialized vectors that are trained (Random init). Table 4.1 shows that the training of the embeddings is beneficial and allows achieving significantly higher recognition scores. However, the usage of static pretrained embeddings also reaches reasonable accuracy while dramatically lowering the time needed for the network training.

	Word emb notrain				**Word emb train**				**Random init**			
	Prec	**Rec**	**F1**	**ACC**	**Prec**	**Rec**	**F1**	**ACC**	**Prec**	**Rec**	**F1**	**ACC**
en	93.0	89.7	91.3	90.2	96.1	93.9	95.0	94.4	96.6	96.3	96.4	96.3
de	95.3	94.8	95.1	95.0	97.0	96.9	96.9	96.8	96.6	96.3	96.4	96.3
es	98.7	98.1	98.4	98.3	99.9	99.9	99.9	99.9	99.9	99.9	99.9	99.9
it	88.8	86.7	87.8	86.9	91.9	91.6	91.7	90.7	91.5	91.2	91.3	90.6
avg	94.0	92.3	93.2	92.6	96.2	95.6	95.9	95.5	96.2	95.9	96.0	95.8

Table 4.1: Results of the single-label classification experiments [in %]

Multi-label Classification

Table 4.2 shows the results of our network in the multi-label scenario. We can summarize the results in this table in a similar way as the previous one for the single-label classification. The training of the embeddings improves the obtained classification results. However, the training of randomly initialized vectors has worse results than the fine-tuned word2vec vectors. The best obtained F-measure 86.8 % is, as in the previous case, for Spanish using word2vec initialized embeddings with a further training.

	Word emb notrain			Word emb train			Random init		
	Prec	**Rec**	**F1**	**Prec**	**Rec**	**F1**	**Prec**	**Rec**	**F1**
en	84.3	62.7	71.9	85.4	89.2	82.2	83.6	75.1	79.2
de	84.2	69.8	76.3	87.5	81.2	84.2	86.5	77.3	81.6
es	90.4	77.1	83.2	89.4	84.3	86.8	89.4	81.5	85.3
it	84.9	68.4	75.8	86.5	81.2	83.8	85.2	77.8	81.3
avg	86.0	69.5	76.8	87.2	81.5	84.3	86.2	77.9	81.9

Table 4.2: Results of the multi-label classification [in %]

Word Similarity Experiment

To demonstrate the quality of the linear transformation of semantic spaces, I present Table 4.3 which shows 10 most similar words to the English word “accident” across all languages based on the cosine similarity. These words are mainly in English when word2vec initialization without any training is used (the first column). Further training of the embeddings (middle column) causes that also German and Spanish words with a similar meaning are shifted closer to the word “accident” in the embedding space. On the other hand, when training from randomly initialized vectors, the ten most similar words have often quite a different meaning. However, as shown in the classification results, this fact has nearly no impact on the resulting F-measure. We can conclude that word2vec initialization is not necessary for the classification task. This table further shows that the similarity between Germanic (English and German) languages is clearly visible.

Word emb notrain		Word emb train		Random init	
word	**cos sim**	**word**	**cos sim**	**word**	**cos sim**
accidents	0.860	accidente	0.685	ruehe	0.248
incident	0.740	unglück (*de*, misfortune)	0.632	bloccando (*es*, blocking)	0.239
accidente (*es*, accident)	0.600	estrelló (*es*, crashed)	0.609	compelled	0.236
incidents	0.574	accidents	0.599	numerick	0.219
accidentes (*es*, accidents)	0.546	geborgen (*de*, secure)	0.585	fiduciary	0.217
disaster	0.471	absturz	0.584	barriles (*es*, barrels)	0.216
explosions	0.461	unglücks (*de*, misfortunes)	0.576	andhra	0.214
incidence	0.452	abgestürzt (*de*, crashed)	0.567	touring	0.212
personnel	0.452	trümmern (*de*, rubble)	0.560	versicherers (*de*, insurers)	0.209
unfall (*de*, accident)	0.450	unglücksursache (*de*, ill cause)	0.551	oppositioneller (*de*, oppositional)	0.203

Table 4.3: Ten closest words to the English word “accident” based on the cosine similarity; English translation in brackets including the language of the given word.

Table 4.4 shows 10 most similar words to the English word “czech” using the cosine similarity. The table is very similar to the previous one. For instance, if we take a look at the *Word emb train* column, we observe that there is (as in the previous case) a significant decrease of the cosine similarity. However on the other hand, some new words, which are more related to the word “czech”, are included. The inapplicability to find similar words of a random initialization has been confirmed. It is worth noting that although the Czech language is not a part of our corpus, some Czech words (praha, dnes, fronta) are also included due to the Czech citations available.

Word emb notrain		**Word emb train**		**Random init**	
word	**cos sim**	**word**	**cos sim**	**word**	**cos sim**
czechoslovakia	0.757	czechoslovakia	0.399	festakt (*de*, ceremony)	0.273
slovakia	0.634	praga (*es*, prague)	0.335	val	0.250
polish	0.569	republic	0.329	provence	0.235
hungary	0.539	brno (*cz*, brno - czech city)	0.315	sostiene (*es*, hold)	0.222
hungarian	0.537	slovak	0.314	larry	0.216
prague	0.533	praha (*cz*, prague)	0.313	köpfigen (*de*, headed)	0.212
slovak	0.509	dnes (*cz*, today)	0.307	überschreiten (*de*, exceed)	0.206
praha (*cz*, praha)	0.509	checa (*es*, czech)	0.307	aktienindex (*de*, share index)	0.205
austrian	0.506	fronta (*cz*, queue)	0.304	councils	0.205
lithuanian	0.496	tschechoslowakei (*de*, czechoslovakia)	0.297	bancario (*it*, banking)	0.205

Table 4.4: Ten closest words to the word "czech" based on the cosine similarity; English translation in brackets including the language of the given word.

4.3.2 Conclusion

The proposed method builds on the popular convolutional networks. A simple yet efficient extension has been added that allows using one network for classifying text documents in more languages.

We evaluated our method on four languages from the Reuters corpus in both multi- and single-label classification scenarios. We showed that the proposed approach is efficient and the best obtained F-measure in the multi-label scenario reaches 84 %. Another contribution of this approach is also that no language identification is needed as in the case of the use of single networks. For detailed information see our papers [34] and [35].

Chapter 5

Dialogue Act Recognition

This chapter first introduces the dialogue act (DA) recognition task and gives an overview of the related work. Finally, it explores the multilingualism in this area. The multilingualism is meant to be a possibility to process multi-lingual dialogue acts. The main emphasis is put on cross-lingual approaches such as semantic space transformation.

5.1 Dialogue Act Recognition Task

Dialogue is a collection of utterances. An utterance is a small unit of speech and it has an unambiguous relation to the speaker. It can be a single word, a sentence or even several sentences. Dialogue act is a function of an utterance in dialogue – it assigns a label to the utterance.

The goal of the recognition task is to classify the utterances (i.e. find the function which assigns a correct label). Unfortunately, there are many DA corpora annotated according to different annotation schemes and thus they contain different DA labels[1]. Despite the diversity of datasets the most common dialogue act labels are very similar (e.g. `QUESTION`, `GREETING`, `RESPONSE` or `STATEMENT`).

Besides these general DA labels, there are many sub-labels relying on the nature of the dialogue (e.g. Figure 5.1). Each utterance has usually a single DA label.

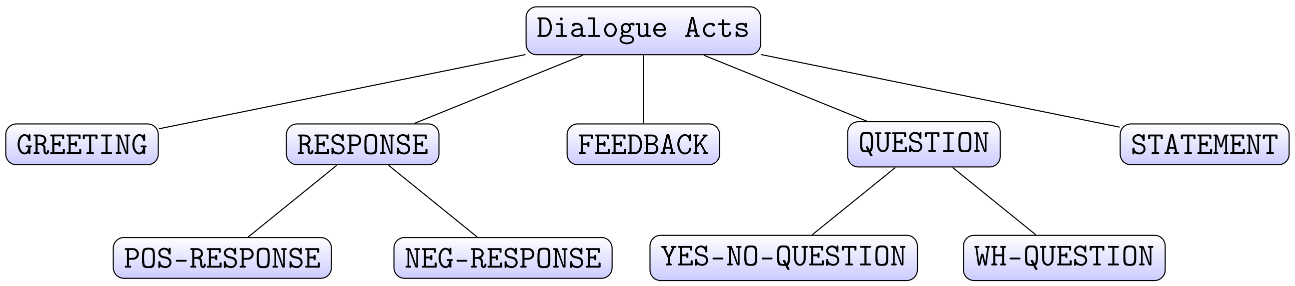


Figure 5.1: Example of the possible structure of dialogue act labels

Very often, the dialogues include parts such as *hmm*, *uh*, and so on, because an utterance is associated with a spoken, not a written, form of dialogue. It is not possible to ignore these

[1]However, relevant efforts towards a standardization of DA annotations have been made [36]

part of speech because *hmm* might refer to the form of acknowledgment. A fragment of the dialogue is presented in Table 5.1

Speaker	Dialogue Act	Utterance
A	YES-NO-QUESTION	So do you go to college right now?
B	**YES-ANSWER**	**Yeah**,
B	**STATEMENT**	**it's my last year.**
A	DECLARATIVE-QUESTION	You're a, so you're a senior now?
B	**YES-ANSWER**	**Yeah**,
B	**STATEMENT**	**I'm working on my projects trying to graduate.**
A	APPRECIATION	Oh, good for you.
B	**BACKCHANNEL**	**Yeah,**
A	APPRECIATION	That's great.
A	YES-NO-QUESTION	um, is, is N C University is that, uh, State,
B	**STATEMENT**	**N C State,**
A	SIGNAL-NON-UNDERSTANDING	What did you say?
B	**STATEMENT**	**N C State,**

Table 5.1: Fragment of a labeled conversation (Switchboard corpus) [49]

5.2 Dialogue Act Recognition Related Work

Traditional DA methods usually create a complex hand-crafted features, for instance, lexical and syntactic information, dialogue history or prosodic information. Prosodic information is often used to provide additional clues to recognize DAs based on intonation, tone, and rhythm (see e.g [46]).

A statistical approach for dialogue acts modeling with a Hidden Markov model is presented by Stolcke et al. in [49]. The proposed probabilistic model which combines decision trees and neural network and it is able to detect and predict dialogue acts based on a dialogue act history and lexical cues such as n–grams.

Nowadays, DA recognition is often tackled with simple word features and sophisticated neural models including popular CNNs and an LSTM. The features can be represented by word embeddings (e.g. provided by word2vec [37])

Another approach using only raw word forms as an input is presented in [6]. The LSTM with word2vec embeddings is used to model the DA structure as well as for DA prediction with very good results.

Duran et al. [12] propose new features called "probabilistic word embeddings" which are based on word distribution across DA-set. The experiments show that these features perform slightly better than word2vec.

Authors in [44] relate a sentiment and dialogues by adding sentiment information and they have improved the task success rate.

The above mentioned approaches are mainly evaluated on English language using Switchboard (SwDA) [28] or Meeting Recorder Dialogue Act (MRDA) [45] corpora. Some methods experimented with Czech [30] or German languages (Verbmobil corpus [25]) though.

The advantage of the last-mentioned Verbmobil corpus is its multilinguality, so there are large enough annotated dialogues in English and German within. It would be possible to unify two foreign-language datasets, but it is hard mainly due to the different annotation scheme. For the research in the multi-lingual DA field, the Verbombil is the most reasonable choice.

5.3 Multi-lingual DA Recognition

This section describes the proposed multi-lingual and cross-lingual experiments on the Verbmobil corpus. The difference between the multi- and cross-linguality lies in training process. In the multi-lingual scenario, we usually train one general model with all available languages. On the other hand, the cross-lingual model is trained on a single pivot language and a transformation method is used to project other languages onto the pivot language.

I describe two approaches which I employed to achieve the multi- and cross- linguality. Before getting to that, though, I want to emphasize the DA context. We consider a DA history (i.e. one or more previous utterances) as a significant feature, because the utterances arise by the context (e.g. the initial utterance with no history is probably `GREETING` or the answer is expected right after question). The context (in the form of the DA history) is included in both methods using teacher forcing [31]. Teacher forcing is a strategy for training recurrent neural networks that uses model output from a previous time step as an input.

A CNN and the Bidirection LSTM have been used as classifiers and word2vec embeddings have been employed to encode word semantics and they are used as a text representation. For both multi- and cross- lingual scenario the proposed models are very similar. The only difference is in the size of a vocabulary and the number of word vectors.

5.3.1 Multi-lingual Model

The multi-lingual model is able to recognize DAs in arbitrarily many languages but it is necessary to retrain the model when a new language is added. There is also no need to perform a semantic space transformation. The ideas behind this model are the same as in the case of the multi-lingual document classification presented in the previous chapter in Section 4.3 Multi-lingual Document Classification.

5.3.2 Cross-lingual Model

The cross-lingual model relies on a semantic space transformation. It is indeed possible to transform the lexical semantic space of any language so that the word representations of similar concepts in different languages are close.

Based on our previous work [35] the canonical correlation analysis (CCA) method has been chosen. It is a technique for multivariate data analysis and dimensionality reduction, which quantifies the linear associations between a pair of random vectors. It can be used for a transformation of one semantic space to another. The model is trained on a single pivot language. The test examples from any language are then projected onto the target pivot language. It thus allows classifying DAs in any language from within the transformed semantic space. Retraining the DA recognition model is not necessary when a new language is considered.

5.3.3 CNNs Approach

We use two CNN networks with different configurations. The first one is the model presented in the previous chapter (see Figure 4.3) where it was used for document classification. The size of the convolutional kernels has been modified to adapt them to the dialogue acts domain. In such a domain, we usually work with much shorter inputs so we use a smaller kernel – (4, 1). All other parameters have not been changed.

The second configuration follows Kim's [29] approach (This architecture was explained in Section 4.2).

So in brief, as in the document classification task, the very same classifier has been employed. The only thing that has changed is the input set of documents.

5.3.4 Bi-LSTM Approach

This approach expoits a Bidirectional LSTM. The representation of the input and the embedding layer are the same as for the CNNs[2]. The core of this model is the Bi-LSTM layer with 100 units (i.e. 200 units in total for both directions). The word embedding representation of the input is fed into the Bi-LSTM layer, which outputs a single vector of 200 dimensions. This vector is then concatenated with the predicted dialogue act class of the previous sentence encoded in the form of a one-hot-vector. If the dialogue act has no history (e.g. initial dialogue act), a zero vector is used. The output layer has a softmax activation function. The architecture is depicted in Figure 5.2.

5.3.5 Experiments

Multi-lingual Verbmobil Corpus

As I indicated above, this Verbmobil corpus is composed of English, German and Japanese dialogues[3]. The dataset is annotated with 42 dialogue acts, which are grouped into the 16 following classes: *feedback, greet, inform, suggest, init, close, request, deliberate, bye, commit, thank, politeness formula, backchannel, introduce, defer* and *offer*. The corpus is very unbalanced. In both languages, there are four dominant DAs (namely *feedback, suggest, inform, request*) which represent almost 80 % of the corpus size.

Experimental Set-up

Word2vec vectors trained on the English and German Wikipedia have been used to initialize the word embeddings in our models. Sentences are truncated or padded to 15 words in all experiments (i.e. input length W = 15). The vocabulary size is set to 10,000.

The evaluation of all models has been made with and without information from the dialogue history, which consists of the dialogue act that has been predicted at the previous sentence. Accuracy metric and Macro F1 score are used to measure the quality of the classifier. Since the corpus is unbalanced, the Macro F1 score is more relevant. The results are averaged based on the 10 runs.

[2] $|V|$ refers to the vocabulary size, W to the input length and EMB is a word vector dimension. C is a set of DA labels

[3] Only English and German dialogues are annotated with DA labels though.

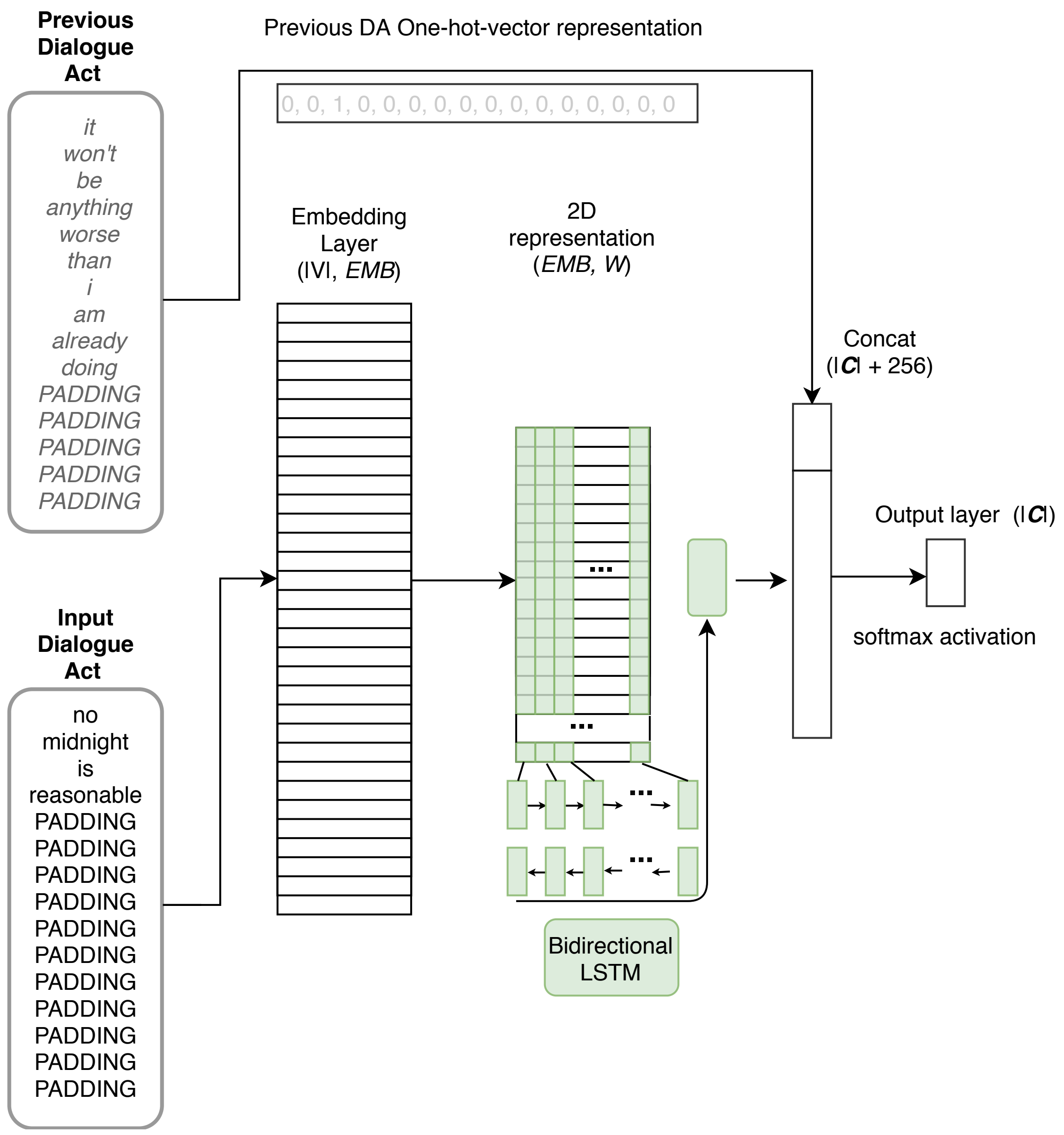

Previous Dialogue Act
it won't be anything worse than i am already doing PADDING PADDING PADDING PADDING PADDING
Previous DA One-hot-vector representation
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
Embedding Layer (|V|, EMB)
2D representation (EMB, W)
Concat (|C| + 256)
Output layer (|C|)
softmax activation
Input Dialogue Act
no midnight is reasonable PADDING PADDING PADDING PADDING PADDING PADDING PADDING PADDING PADDING PADDING PADDING
Bidirectional LSTM

Figure 5.2: Bi-LSTM DA classifier architecture

Multi-lingual Results

This series of experiments shows results of the multi-lingual model. Table 5.2 reports the performance of the models with static word2vec embeddings while Table 5.3 presents the results with fine-tuned embeddings. These tables show that, generally, fine-tuning word2vec embeddings does not bring any improvement for DA recognition.

Cross-lingual Results

Table 5.4 shows the cross-lingual results. The scores of the cross-lingual model are significantly lower than the scores of the multi-lingual methods. The best reported accuracy is obtained by the Bi-LSTM network and it is close to 60 % when we use the German part of the corpus for training (pivot language) and the English dataset for testing. Low F1 score oc-

		CNN				CNN – Kim				Bi-LSTM			
		With History		**No History**		**With History**		**No History**		**With History**		**No History**	
Train	Test	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
en	en	72.1	58.9	72.2	58.4	74.5	**65.8**	74.3	65.1	**74.9**	60.5	74.1	59.9
de	de	72.5	**60.8**	71.8	58.2	71.9	57.5	70.8	56.6	**74.3**	59.3	73.6	59.4
en+de	de	72.0	59.9	71.2	57.7	70.9	57.5	71.1	54.6	**74.3**	**61.7**	73.2	60.6
en+de	en	70.3	55.1	70.0	55.5	71.4	57.1	70.7	**58.6**	**72.8**	58.5	72.6	57.4

Table 5.2: Multi-lingual DA recognition with static word2vec embeddings

		CNN				CNN – Kim				Bi-LSTM			
		With History		**No History**		**With History**		**No History**		**With History**		**No History**	
Train	Test	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
en	en	72.2	**69.2**	72.2	68.4	**73.7**	68.4	72.1	59.1	73.5	67.2	72.7	67.2
de	de	72.7	59.2	71.7	57.7	72.1	59.1	72.6	**60.4**	**74.9**	57.2	74.3	59.1
en+de	de	71.8	**60.8**	70.8	58.9	70.8	58.4	71.7	58.8	**72.7**	58.2	71.4	58.3
en+de	en	**69.2**	61.2	68.6	58.6	69.6	61.2	68.7	60.2	68.5	60.1	**69.2**	**63.1**

Table 5.3: Multi-lingual DA recognition with fine-tuned word2vec embeddings

curred because of the poor results of infrequent DAs, which do not impact much the accuracy values. This table further shows that dialogue history slightly helps for DA recognition and that the Bi-LSTM significantly outperforms the two other CNN models.

The lower results for the English $\rightarrow$ German direction can be explained by the significantly smaller corpus size for training (see Table 5.5).

		CNN				CNN – Kim				Bi-LSTM			
		With History		**No History**		**With History**		**No History**		**With History**		**No History**	
Train	Test	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
en	de	30.7	11.9	**34.3**	13.9	31.5	14.5	31.2	15.4	34.0	16.4	34.0	**17.0**
de	en	55.1	26.4	54.4	25.5	53.9	28.3	53.0	27.4	**58.6**	**37.1**	57.5	33.7

Table 5.4: Cross-lingual DA recognition based on CCA transformation

	English		German	
unit	Training	Testing	Training	Testing
dialogue #	6 485	940	15 513	622
DA #	9 599	1 420	32 269	1 460
word #	79 506	11 086	297 089	14 819

Table 5.5: Verbmobil corpus statistical information

5.3.6 Conclusion

We have compared and evaluated two different CNN configurations and one Bi-LSTM on the Verbmobil corpus with English and German DAs. It has been shown that the results of our multi-lingual model significantly outperforms the cross-lingual approach. The advantage of the multi-lingual model is that it does not need language detection. However, the multi-lingual model is less flexible and may not scale easily to many languages, because retraining is necessary when adding new languages.

Another interesting observation is that fine-tuning of word2vec embeddings does not bring any improvement in this task. The dialogue history is beneficial for DA recognition in almost all cases and that the best neural classifier is the Bi-LSTM network.

The presented approaches may be improved in many ways, for instance by exploiting contextual word embeddings, transformer-based encoders and by annotating more languages. In the short term, I plan to use the Bi-LSTM classifier in the document classification task to evaluate its quality on a different dataset.

Chapter 6

Aims of the Doctoral Thesis

As reported in previous chapters, the main emphasis of my research will be put on deep learning methods. I proved that they are beneficial in three important NLP tasks. I also showed that the very similar classifier (with minimal necessary changes of parameters) can be easily reused with a success in different domain (i.e. relatively long document categorization and very short utterances of a dialogue).

I will divide the doctoral thesis into three blocks, namely OCR for Historical documents, document classification and dialogue acts recognition. Either of them has been covered within this report, with a support of relevant publications. All mentioned classification tasks will be tackled by deep neural network models. Although the OCR in historical documents (presented in Chapter 3) and NLP have relatively little in common, I proved that some relations between them can be found (e.g. implicitly learn language model or seq2seq and language model based error correction). A correction of the wrong recognized text is the final phase of the OCR, which has not been covered yet. Nevertheless, there is a room to use NLP models to address this issue.

In my future research activity, I also would like to concentrate on the multi-linguality of the presented NLP methods, primarily multi-lingual dialogue acts recognition and document classification fields. This area has a big research potential, just because there is a lack of the multi-lingual corpora.

The following list of aims of the doctoral thesis implies from the above-mentioned.

1. Proposition of the modern deep learning OCR methods for historical document processing
2. Design and use deep neural networks for text categorization with a multi- and cross-lingual application
3. Proposing and applying deep learning models in the multi-lingual dialogue act recognition field

Chapter 7

Conclusions

The presented report has three main contributions. First, it provides the theoretical summary of popular deep learning methods. Second, it maps my research and my publication activity so far. Finally, it presents the aims of the doctoral thesis and the goals for further research.

In Chapter 2, I described the popular approach called "deep learning". Some of the presented methods have been used in my recent research, as evidenced by the other chapters. This theoretical basis fully reflects the current trends and research directions.

Chapter 3 deals with the OCR of historical documents. The interesting topics are the implicit language model learned by RNN in OCR system and seq2seq model for error correction.

Chapters 4 and 5 deal with document classification and dialogue act recognition. These research fields are handled by very similar classifiers and also many other aspects are in common.

In the previous, I determined a list of goals I want to accomplish in my doctoral thesis. I lean on my research activities as well as the theory described in Chapter 2 in particular. Considering my work so far, the reported aims are not beyond the scope of my research topic and they are worth exploring more deeply.

Bibliography

[1] Massih Amini, Nicolas Usunier, and Cyril Goutte. Learning from multiple partially observed views-an application to multilingual text categorization. In *Advances in neural information processing systems*, pages 28–36, 2009.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Thomas M Breuel. The ocropus open source ocr system. In *Document Recognition and Retrieval XV*, volume 6815, page 68150F. International Society for Optics and Photonics, 2008.

[4] Thomas M Breuel. High performance text recognition using a hybrid convolutional-lstm implementation. In *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on*, volume 1, pages 11–16. IEEE, 2017.

[5] Tomáš Brychcín. Linear transformations for cross-lingual semantic textual similarity. *arXiv preprint arXiv:1807.04172*, 2018.

[6] Christophe Cerisara, Pavel Kral, and Ladislav Lenc. On the effects of using word2vec representations in neural networks for dialogue act recognition. *Computer Speech & Language*, 47:175–193, 2018.

[7] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

[8] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[9] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[10] Christian Clausner, Stefan Pletschacher, and Apostolos Antonacopoulos. Efficient ocr training data generation with aletheia. *Proceedings of the International Association for Pattern Recognition (IAPR), Tours, France*, pages 7–10, 2014.

[11] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. *arXiv preprint arXiv:1606.01781*, 2016.

[12] Nathan Duran and Steve Battle. Probabilistic word association for dialogue act classification with recurrent neural networks. In *International Conference on Engineering Applications of Neural Networks*, pages 229–239. Springer, 2018.

[13] Shivansh Gaur, Siddhant Sonkar, and Partha Pratim Roy. Generation of synthetic training data for handwritten indic script recognition. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 491–495. IEEE, 2015.

[14] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[15] YYRTG Glewis, D David, and F Li. A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 2004.

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[17] Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 347–352. IEEE, 1996.

[18] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[19] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868, 2009.

[20] Alex Graves and Jürgen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552, 2009.

[21] Maya R Gupta, Nathaniel P Jacobson, and Eric K Garcia. Ocr binarization and image preprocessing for searching historical documents. *Pattern Recognition*, 40(2):389–397, 2007.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[23] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[24] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014.

[25] Susanne Jekat, Alexandra Klein, Elisabeth Maier, Ilona Maleck, Marion Mast, and J Joachim Quantz. Dialogue acts in verbmobil. 1995.

[26] Rie Johnson and Tong Zhang. Supervised and semi-supervised text categorization using lstm for region embeddings. *arXiv preprint arXiv:1602.02373*, 2016.

[27] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, April 2017.

[28] D Jurafsky, E Shriberg, and D Biasca. Switchboard swbd-damsl shallow-discourse-function annotation coders manual (1997). *URL http://web. stanford. edu/˜ jurafsky/ws97/manual. august1. html*.

[29] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[30] Pavel Král, Christophe Cerisara, and Jana Klecková. Combination of classifiers for automatic recognition of dialog acts. In *Ninth European Conference on Speech Communication and Technology*, 2005.

[31] Alex M Lamb, Anirudh Goyal Alias Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609, 2016.

[32] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[33] V Margner and Mario Pechwitz. Synthetic data for arabic ocr system development. In *Document Analysis and Recognition, 2001. Proceedings. Sixth International Conference on*, pages 1159–1163. IEEE, 2001.

[34] Jiri Martinek, Ladislav Lenc, and Pavel Kral. Neural networks for multi-lingual multi-label document classification. In *27th International Conference on Artificial Neural Networks (ICANN 2018)*, volume 11139 LNCS, pages 73–83, Rhodes, Greece, October 4-7 2018. Springer International Publishing.

[35] Jiri Martinek, Ladislav Lenc, and Pavel Kral. Semantic space transformations for cross-lingual document classification. In *27th International Conference on Artificial Neural Networks (ICANN 2018)*, volume 11139 LNCS, pages 608–616, Rhodes, Greece, October 4-7 2018. Springer International Publishing.

[36] Stefano Mezza, Alessandra Cervone, Evgeny Stepanov, Giuliano Tortoreto, and Giuseppe Riccardi. Iso-standard domain-independent dialogue act tagging for conversational agents. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3539–3551, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[37] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[38] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[39] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[40] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[41] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and SS Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):92, 2018.

[42] Ekraam Sabir, Stephen Rawls, and Prem Natarajan. Implicit language model in lstm for ocr. In *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on*, volume 7, pages 27–31. IEEE, 2017.

[43] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2017.

[44] Weiyan Shi and Zhou Yu. Sentiment adaptive end-to-end dialog systems. *arXiv preprint arXiv:1804.10731*, 2018.

[45] Elizabeth Shriberg, Raj Dhillon, Sonali Bhagat, Jeremy Ang, and Hannah Carvey. The icsi meeting recorder dialog act (mrda) corpus. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue at HLT-NAACL 2004*, 2004.

[46] E. Shriberg *et al.* Can prosody aid the automatic classification of dialog acts in conversational speech? In *Language and Speech*, volume 41, pages 439–487, 1998.

[47] Fotini Simistira, Adnan Ul-Hassan, Vassilis Papavassiliou, Basilis Gatos, Vassilis Katsouros, and Marcus Liwicki. Recognition of historical greek polytonic scripts using lstm networks. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 766–770. IEEE, 2015.

[48] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

[49] Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373, 2000.

[50] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[51] Yaodong Tang, Yuchen Huang, Zhiyong Wu, Helen Meng, Mingxing Xu, and Lianhong Cai. Question detection from acoustic features using recurrent neural network with gated recurrent unit. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6125–6129. IEEE, 2016.

[52] Jaya Thomas, Sonia Thomas, and Lee Sael. Feature versus raw sequence: Deep learning comparative study on predicting pre-mirna. *arXiv preprint arXiv:1710.06798*, 2017.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[54] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

[55] Zhiwei Zhao and Youzheng Wu. Attention-based convolutional neural networks for sentence classification. In *INTERSPEECH*, pages 705–709, 2016.