1 Motivation and Organization of the Thesis

Even if the amount of data increases inexorably due to the permanent collection of online data, and there are always new and better methods to derive good forecasts from big data, it is still difficult for algorithms to extract information from unstructured data. The amount of text data does not increase as quickly as the amount of structured user data. However, it should not be ignored that the written word still has an immensely high -if not the highest - information density for the human mind. Text is the primary and most accurate way to transfer and store complex information. In a nutshell: while a machine and algorithms think in numbers, humans still think in words and text. A major challenge of machine learning will be to extract and link information from unstructured data such as text.

A whole field of computer science is already addressing the topic of text data. This is Natural Language processing and Information Retrieval (see chapter 2). In the recent past, especially probabilistic approaches have been considered promising by researchers (see e.g. [Manning, 1999]). However, the success of methods depends greatly on the text data to which they are applied and on the desired output. It is therefore equally important to address the methods and models used, as well as describing the use case in detail, when conducting research in this area. Winter et al. for example used regulatory documents and guidelines to characterized them via KNN [Winter, 2017]. In this work, I want to follow Winter's work thematically. I will examine related as well as new, completely different methods and approaches to characterize regulatory documents.[1]

In this work I adhere to the paradigms of Dejong's 1979 work in the field of natural language processing. He was the first researcher in this area to move away from the story-specific approach by testing the accuracy of his program to develop a more robust model [DeJong, 1979]. I even go one step further and optimize my model on completely different data than those it will be tested on later. By doing so I try to reduce the overfitting of the model to the chosen topic and develop a universal tool for text classification. Specifically, I will train my approach for classifying text documents on a relatively large set of textbook chapters. I will optimize the models and their hyperparameters and test the optimized models with a smaller set of regulatory documents from EUROSTAT.

[1] When comparing the two works, however, note the difference in the topic of the documents analyzed.

2 Natural Language Processing and Information Retrival

The reason for the current relevance of the topic NLP is obvious. But what does the concept of natural language processing actually cover and into which research field is it to be classified? It is worth looking at how NLP has evolved over the years and where it originated, in order to better understand what this area encompasses and what it does not. Since the origins of NLP lie in computer science, especially in Artificial Intelligence, NLP and AI shares the same approaches. For an short review of the motivations of AI, see also Chapter 3.2.1.

“Between about 1960 and 1985, most of linguistics, psychology, artificial intelligence, and natural language processing was completely dominated by a ***rationalist*** approach” [Manning, 1999, p. 4]. In other words, procedures for NLP were completely static, fix-programmed solutions for processing text data. Books like [Noble, 1988] followed this approach until the late 80s. Whereby the enthusiasm gradually, just like for artificial intelligence itself, decreased and arose cyclical.

The second school of thought mentioned by Manning and Schütz is the ***empiristic*** approach. This is based upon the assumption, that one “can learn the structure of language by specifying a general model and then learning the values of the parameters by applying statistical modeling and machine learning methods to large amount of observed language” [Martinez, 2010, p. 253]. What finally led to the breakthrough of NLP in its current form is what Manning defines as statistical NLP. I.e. all quantitative approaches for automatization NLP, which includes probabilistic modeling, as well as information theory, and linear algebra. This thesis will focus on methods from this approach.
Some of this work could even in fact fall a step further. Deep learning (see Chapter 3.2.1) was indeed invented already in 1999 at the time of the division of the definition Mannings in its basic outlines, but received still far less attention than today. One could therefore call NLP via deep learning a subcategory of the empirical approach in its own right, and possibly even list ”deep NLP” alongside Statistical NLP.

Statistical NLP considers recurring patterns and structures always in the context of certain texts and does not examine the whole language as such. This approach is no new invention by NLP researchers, but has been used by linguists even prior to computer science (see [Harris, 1951]). A collection or body of texts is called ***corpus***, which means ”body” in Latin. In the course of this work we will also speak of several corpora, which refers to several collections of texts.

In this thesis so called topic models are studied and applied. David M. Blei defines topic models “as algorithms for discovering the main themes that pervade a large and otherwise unstructured collection of documents. Topic models can organize the collection according to the discovered themes”[Blei, 2012]. This definition may therefore apply to both clustering and classification algorithms, even if many authors will refer to clustering algorithms in the context of topicmodels. You will find a brief introduction to clustering and classification models in the use case in Chapter 4.

Another important term in the research area of this thesis is information retrieval (IR). IR means to give access to a subset of documents of a corpus, which are relevant to an user's query. This field deals with computer-aided searches for complex contents and belongs to the fields of information science, computer science and computational linguistics. According to David Grossmann et al. IR refers to a search that might cover information of any kind, e.g. text data as well as video-, image-, sound data, or even DNA sequences. The field where LDA and IR overlap is document retrieval [Grossmann, 2004]. Thus, most procedures we are discussing in this thesis might not solely fall into the topic NLP but also IR. This small excursion in this direction is intended to give the reader a better understanding of how linked this topic is and in which intersection it is located.

In addition, there is a further differentiation of methods concerning natural language processing. One can also distinguish text interpretation with regard to the tasks one intends to accomplish with it. [Jacobs, 1993] distinguishes between three common tasks:

1. *Information Retrieval*: The task to find a subset of a corpus, which contains information concerning a user's query.

2. *Data extraction*: The task to bring texts of a corpus of a particular domain into a pre-defined key structure, suitable for use in a traditional database.

3. *Text categorization*: Separating documents of a corpus into meaningful groups.

The tasks dealt with in this paper can perhaps best be assigned to the last area. On the one hand, I specifically target the task text categorization by dividing a corpus of documents into a predefined number of groups. On the other hand, I try to assign new documents to these groups.

In Chapter 2 I will discuss the evolution of natural language processing.

The different approaches and methods will be summarized and the reader will be given a general overview over the context of this work.
Chapter 3 presents the models used and describes the form of application in detail. The two methods used here do not only differ in terms of their underlying models, but also have different tasks.

3 Discussed Models

This section will discuss the two Topic modeling approaches which will be studied in this Thesis. The aim of both procedures is to assign one or more topics to different documents. Even if the vocabulary and the notation are similar for both approaches, the notation should be resumed at the beginning of the description of each model. The basic structural notation of the data consists of the following variables.

A collection of documents is called corpus $D = (\mathbf{w}_1, \ldots, \mathbf{w}_M)$. It consists of M documents $\mathbf{w} = (w_1, \ldots, w_N)$ which represents each of the N words w_i in the vocabulary. These words are vectors of length V. V refers to the length of a vocabulary which holds all the words occurring in the corpus. The vector for a specific word w_i contains all 0 except for index $j \in \{1, ..., V\}$ which represents this very one word in the vocabulary. This notation may be extended through the addition of indices for documents, but this is done neither here nor in the standard literature on topic models due to its unnecessary complexity.

3.1 LDA Model

Latent Dirichlet Allocation is a Bayesian approach and is often associated with hierarchical models [A. Gelman, 2014]. This idea is based on the representation of exchangeable random variable as mixtures of distributions as discussed by de Finetti. Given that documents $\mathbf{w}$ and words w_i in each document - both considered as random variables in this setting - are exchangeable in such a way, a mixed model such as the LDA model is appropriate [Blei, 2003].

The following notation is used in conjunction with the LDA model. Let z_j be the topics with $j \in \{1, \ldots, k\}$. In the LDA setting we assume for every topic z_j there is a term distribution

$$\beta_j \sim Dir(\delta)$$

We further assume each document w has a distribution of topics.

$$\theta \sim Dir(\alpha)$$

Then each word w_i of $\mathbf{w}$ is generated by the following process:

1. Choose $z_i \sim Mult(\theta)$
2. Choose $w_i \sim Mult(\beta_i)$ This distribution will be referred to as $p(w_i|z_i, \beta)$

You can summarize this setup in a plate diagram as shown in figure 1. The notation above, which is also used within the diagram, coincides with the notation of [Hornik, 2011].

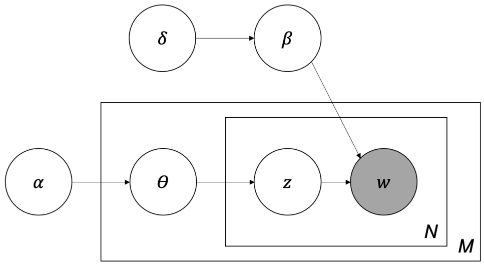

Figure 1: The well-established plate diagram for the standard LDA model extended by the parameter δ. The slightly bigger box represents the generative model of the corporis M documents. The smaller plate represents the iterative generation process of the N words of each document with the aid of the topics. See also "smoothed LDA model" in [Blei, 2003] for comparisons.

In order to estimate the model's parameters, the first step is to calculate posterior distribution, which can be done by dividing the joint distribution by the marginal distribution.

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)} \tag{1}$$

The joint distribution numerator can be derived with the following calculation:

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{i=1}^{N} p(w_i|z_i, \beta)\ p(z_i|\theta) \tag{2}$$

One can obtain the marginal distribution of a document $\mathbf{w}$, by integrating out the parameter θ and summing up the topics z_j. Nevertheless, this expression is intractable.

$$p(\mathbf{w}|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{i=1}^{N} \sum_{z_i} p(z_i|\theta) p(w_n|z_i, \beta) \right) d\theta \tag{3}$$

The literature divides the approaches to calculating posterior distribution into two main categories.[Blei, 2012] distinguishes between-sampling based algorithms and variational algorithms. [Powieser, 2012] lists a total of 6 algorithms that can be used to estimate parameters in the LDA model. This thesis will be confined to the two most cited and most used members of the two main groups. One approach is to simulate the posterior density by iteratively sampling - the so-called Gibbs Sampling method. The second approach is a deterministic method, a modified version of the well-known EM algorithm [AP Dempster, 1977]: the Variational EM algorithm (VEM algorithm) [Wainwright and Jordan, 2008]. In the following two sections the both approaches are roughly outlined to give the reader some insight into the Bayesian inference underlying the algorithms.

3.1.1 Variational EM Algorithm

In the VEM algorithm for the LDA model is a mean field approach which varies the steps E and M of the EM algorithm in a way such that this algorithm becomes solvable. Note that the main problem in calculating marginal distribution lies in deriving the conditional probability of hidden variables in the observed values ('evidence'). Variation in the EM algorithms arises mainly from approximating the directly intractable E step. Rewriting the log of the border density of $\mathbf{w}$ as follows in (4), results in a downward estimation of marginal density given Jensen's inequality.

$$\begin{aligned}
\log p(\mathbf{w}|\alpha,\beta) &= \log \int \sum_z p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)d\theta && (4)\\
&= \log \int \sum_z \frac{p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)q(\theta,\mathbf{z})}{q(\theta,\mathbf{z})}d\theta && (5)\\
&\geq \int \sum_z q(\theta,\mathbf{z})\log p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)d\theta - \int \sum_z q(\theta,\mathbf{z})\log q(\theta,\mathbf{z})d\theta && (6)\\
&= \mathbb{E}_q[\log p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)] - \mathbb{E}_q[\log q(\theta,\mathbf{z})] && (7)
\end{aligned}$$

Here $q(\theta,\mathbf{z})$ is an arbitrary distribution which can be called the variational distribution.

$$q(\theta,\mathbf{z})\hat{=}q(\theta,\mathbf{z}|\gamma,\phi) = q(\theta|\gamma)\prod_{i=1}^{N} q(z_i|\phi_i) \tag{8}$$

The right hand side $L(\gamma,\phi,\alpha,\beta) := \mathbb{E}_q[\log p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)] - \mathbb{E}_q[\log q(\theta,\mathbf{z})]$ is referred to as the 'lower bound'. It can be shown that $\log p(\mathbf{w}|\alpha,\beta) - L(\gamma,\phi,\alpha,\beta)$ is the Kullbak Leibler divergence (D_{KL}) of the true posterior and the variational distribution. From equations (4)-(7) follows that:

$$\log p(\mathbf{w}|\alpha,\beta) = D_{KL}(q(\theta,\mathbf{z}|\gamma,\phi)||p(\theta,\mathbf{z},\mathbf{w}|\alpha,\beta)) + L(\gamma,\phi,\alpha,\beta) \tag{9}$$

Since the marginal distribution is fixed, we conclude that minimizing the KL-divergence is equivalent to maximizing the lower bound (see [Jordan, 1999] and [Wainwright and Jordan, 2008], for details of the derivation of the lower bound see [Blei, 2003]).

$$(\gamma^*, \phi^*) = \underset{\gamma,\phi}{\operatorname{argmin}}\, D_{KL}(q(\theta, \mathbf{z}|\gamma, \phi)||p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)) \tag{10}$$

$$= \underset{\gamma,\phi}{\operatorname{argmax}}\, L(\gamma, \phi, \alpha, \beta) \tag{11}$$

The EM algorithm thus is to use the variational distribution $q(\theta, \mathbf{z}|\gamma^*(\mathbf{w}), \phi^*(\mathbf{w}))$ instead the posterior distribution $p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)$. Now the two steps of the VEM algorithm are:

(1) **E step** Optimize the variational parameters θ and ϕ for every document in the corpus. This can be done analytically by deriving the derivatives of the KL divergence. And set them to zero.

(2) **M Step** Maximize the lower bound using the optimized parameter of the E step with respect to α and β.

3.1.2 Gibbs Sampling

The second method to approximate the posterior distribution is Gibbs sampling, a form of the Monte Carlo method. Instead of calculating the distributions for β and θ, the primary task is to find the posterior distribution over $\mathbf{z}$ given the document $\mathbf{w}$. Gibbs sampling is also known as a Markov Chain Monte Carlo method. The name refers to the simulation process by which a chain of values is simulated whose limiting distribution desirably converges against the true distribution [M. Steyvers, 2006]. (12) shows the distribution, which is sampled from iteratively.

$$p(z_i = j|z_{-i}, w) \propto \frac{n_{-i,j}^{(l)} + \delta}{\sum_t n_{-i,j}^{(t)} + V\delta} \frac{n_{-i,j}^{(d_i)+\alpha}}{n_{-i}^{(d_i)} + k\alpha} \tag{12}$$

$z_i = j$ ... word-topic assignment of word i to topic j
z_{-i} ... vector of word-topic assignments without the entry for word i
$n_{-i,j}^{(l)}$ ... number of times the lth word in the vocabulary is assigned to topic j, not including the assignment for word i
d_i ... document in the corpus which includes word i
δ, α ... parameters of the prior distributions for β and θ

The word-topic distributions $\beta_j^{(l)}$ for the words $l = 1, ..., V$ and topics $j = 1, .., k$ and topic-document distributions $\theta_j^{(d)}$ for the documents $d = 1, ..., D$

and the topics $j = 1, ..., k$ will be of particular interest. (13) and (14) shows the predictive distributions denoted as "estimators".

$$\hat{\beta}_j^{(l)} = \frac{n_{-i,j}^{(l)} + \delta}{\sum_t n_{-i,j}^{(t)} + V\delta} \tag{13}$$

$$\hat{\theta}_j^{(d)} = \frac{n_{-i,j}^{(d_i)+\alpha}}{n_{-i}^{(d_i)} + k\alpha} \tag{14}$$

For derivation and more details regarding the Gibbs sampling procedure see [M. Steyvers, 2006].

3.1.3 Implementation

In this thesis, the implementation of the LDA model and its estimation is mainly based on using the package `topicmodels` of Kurt Hornik. The package `topicmodels` can apply both the VEM algorithm as well as Gibbs sampling in order to fit the model. In addition, the package `tidytext` is used for text structuring and embedding. Whereby there are other packages besides this implementation of the LDA model, `topicmodels` is particularly convenient, because `tidytext` was designed by its developers to work perfectly in combination with `topicmodels` [Silge and Robinson, 2017, p. 89].

3.2 Artifical Neural Networks

Artificial neural networks (ANN) are much more versatile than the LDA model. There are not only various forms of artificial neural networks, but also a very large number of application areas. Much like machine learning procedures in general, also deep learning algorithms are divided into two broad categories: supervised learning, where a superset instance provides the algorithm with the output required to learn, and unsupervised procedures that internally train predefined models to find patterns in the input signals. In this chapter we will focus heavily on the former group of ANNs. Also, this chapter is intended to give the reader an overview of the research on neural networks as well as the background of their development.

3.2.1 Development

Research on ANNs dates back to the 1940s, when [McCulloch and Pitts, 1943] introduced the so called "M-P neuron". Whereby this neuron had only a bivariate input and output, Rosenblatt later extended this idea to a network of M-P neurons, which allowed to set up a simple classification algorithm [Rosenblatt, 1958]. A perceptron in its basic form (single perceptron) is a binary classifier.

Imagine input data of a simple perceptron in the form of a matrix.

$$X = \begin{bmatrix} x_{11} & \dots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nn} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$$

The dependent variable thus is a vector $\mathbf{y} = y_1, \dots, y_n$, with $y_i \in \{0, 1\}$. Consider the lines of the X mtrix as vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$, with $\mathbf{x}_i \in \mathbb{R}^k$. The entries of each of the vectors are weighted with $\mathbf{w} = w_1, \dots, w_k$ with $w_j \in \mathbb{R}$ and aggregated in a function h e.g. a sum.

$$h(\mathbf{x}, \mathbf{w}) = \sum_{j=1}^{k} x_j w_j$$

Using a so called "activation function" h is mapped to the output space, which is in this case $O = \{0, 1\}$. At this point a step function serves as activation function.

$$a \circ h(\mathbf{x}, \mathbf{w}) = \begin{cases} 0 \text{ if } h(\mathbf{x}, \mathbf{w}) \leq 0 \\ 1 \text{ else} \end{cases}$$

The matrix X is passed vector by vector to the percepron and the output is compared with the values for $\mathbf{y}$. During this procedure the weights are iteratively tuned by a simple updating algorithm using the pairs $\mathbf{x}_i$ and y_i.

The algorithm of the simple perceptron is schematically shown in Figure 2. This diagram corresponds to the common representation in education [Mukherjee, 2019], although a horizontal perspective is often chosen.

If this basic perceptron is used in a clever way, designs can be developed that have many different application possibilities. For example, a sigmoid function can be used as an activation function instead of the step function. So the output layer will not project into the $\{0, 1\}$ space, but into a probability space. If you add several nodes with the sigmoid activation function rather than a single output, you basically obtain the architecture that is also called multivariate logistic regression [Bahjat, 2006]. In that case the model can be estimated by an individual calculation of logistic regressions for each node in the output layer. This allows to classify not only two classes, but an arbitrary number (see the network architecture of Figure 3). [2]

Shortly following the publication of these results, which would lay the foundation for later neural networks, ANN research had to suffer a severe setback

[2] Note that in this setting the dependent variable must be one-hot encoded.

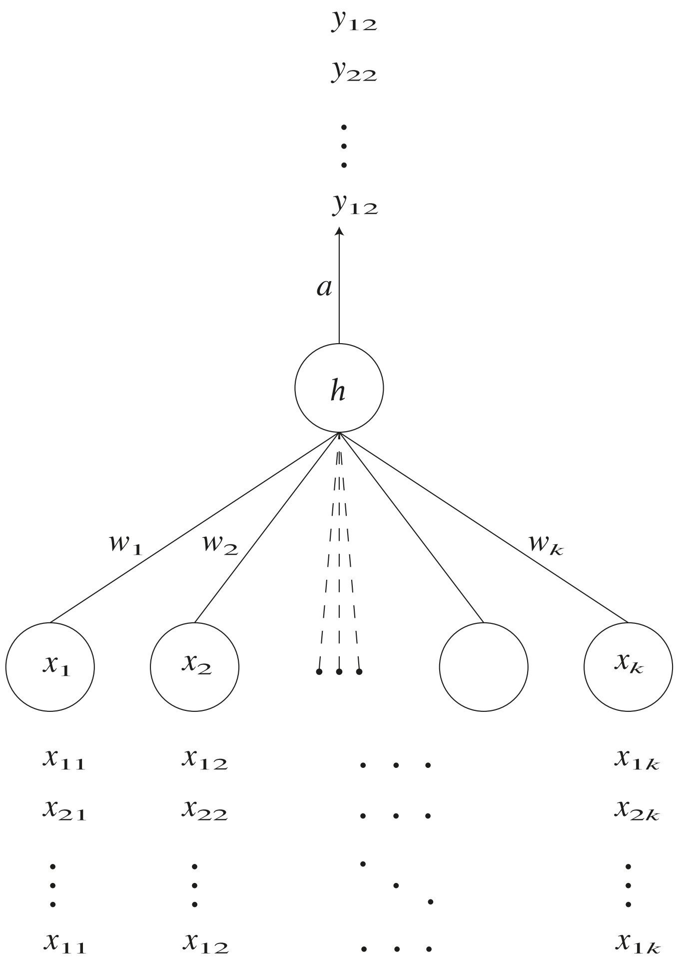

Figure 2: Schematic diagram of a simple perceptron by [Rosenblatt, 1958]

after Minsky and Papert [Minsky and Papert, 1969] were able to prove that perceptrons cannot provide a suitable solution in certain basic scenarios. It is possible to separate two clusters by a hyperplane with the perceptron. However, if the two groups could not be separated completely, the perceptron fails. For instance, it was impossible to find a solution for data generated with an *x-or* function (also called ”exclusive or” function).

Although the researchers Minsky and Papert showed that the perceptron in this form cannot solve the “xor problem”, they argued that extending the simple perceptron to a multi layer perceptron solves this problem, if it was feasible to train this model. Instead of just one input layer and one output layer, an MLP may include an arbitrary number of hidden layers in between. Figure 4 shows the structure of such an network. However, training a model using Rosenblatt’s naive optimization algorithm of the perceptron would not

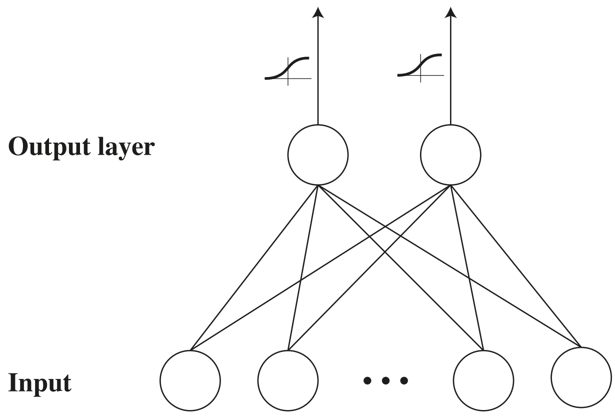

Figure 3: Schematic diagram of an adapted Rosenblatt-perceptron network

have been feasible.

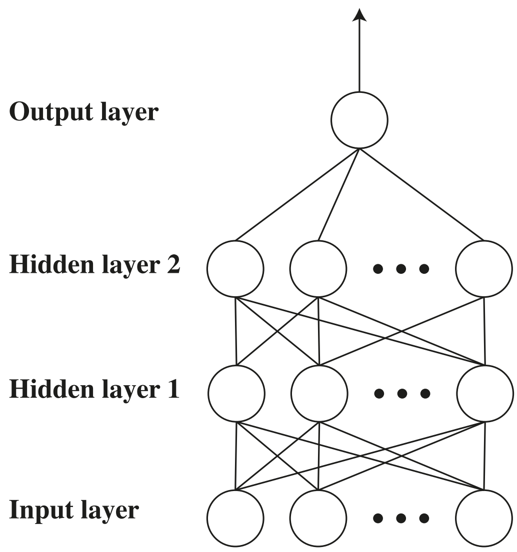

Figure 4: Schematic diagram of an multi layer perceptron (MLP)

3.2.2 Backpropagation

The procedure used to optimize the weights of a multilayer perceptron is called backpropagation. This method was first applied to neural networks by Webos - as part of his dissertation [Werbos, 1974]- and still works the same way to this day. It is important to use a sigmoid function instead of the step function which is used in the perceptron, as this function is differentiable. Assume a random initial distribution of the weights $w_1, \dots, w_d$ for a network with d layers, which are the starting values for the update procedure. Let τ_i be the number of units in layer i. So layer i can be denoted as function f_{w_i} and the whole network as a chain of functions.

$$\hat{y_i} = f_{w_d}(f_{w_{d-1}}(...(f_{w_1}(x_i)))) = f_{w_d} \circ f_{w_{d-1}} \circ \cdots \circ f_{w_1}(x_i) \tag{15}$$

The goal is to update the weights of each layer in sucha way as to minimise a selected loss function $L(w_1, \ldots, w_d)$. A common loss function is for example quadratic loss: $L = \frac{1}{2}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2$
The backpropagation algorithm consist of the 3 following roughly outlined steps.

1. **forward pass:** Calculation of the result y_i for all units at the set weights
2. **backward-sweep:** Each layer's derivatives are now calculated using the chain rule, step by step - starting with the output layer. For the output layer this is:

$$\frac{\partial L}{\partial w_d} = \frac{\partial L}{\partial f_d(x_i)} \frac{\partial f_d(x_i)}{\partial w_d}$$

3. **updating:** Update the weights by an arbitrary optimization algorithm. E.g. steepest descent: $\Delta w_d = \alpha \frac{\partial L}{\partial w_d}$,where α is the learning rate

Thanks to current software, the derivatives no longer have to be calculated for each node by hand, but the standard packages are capable of "symbolic differentiation" for certain network structures, which makes training neural networks comfortable [Chollet, 2018, p. 47].

Note: While Rosenblatt for the perceptron was actually inspired by the biological structure of the neurons in the brain, more complex "networks used by engineers are only loosely based upon biology"[Hecht-Nielsen, 1988].

3.2.3 Implementation

This thesis uses the `Keras R` package to create neural networks, a deep learning API (Application-Programming-Interfac) to deep learning backend engines, designed for `R` by Allaire in 2017. Keras is a so called "model-level" library, designed to set up complex neural net architectures using well arranged, high-level building blocks. As backend-engine the users are provided with TensorFlow, Theano and Microsoft Cognitive Toolkit. Using these backend engines computation may be processed seamlessly via CPU or GPU [Chollet, 2018].

4 Analysis of Gutenberg Data

In this first example, the chapters of individual books are classified, all of which sourced from the freely distributed Project Gutenberg. Project Gutenberg is a provider of over 60,000 free electronic books with the primary aim to “encourage the creation and distribution of eBooks”[Hart,]. The package `gutenbergr` [Robinson, 2018] preprocesses the ebooks text data and downloads it. This convenient package in combination with the free repository allows the analysis of a large number of large text documents with a secure source and a big amount of meta information. For this analysis interesting information is e.g. Gutenberg id as key variable for fast identification of books, title, author and more importantly the categorization of project Gutenberg, the so-called Gutenberg bookshelf. Table 1 lists all this information for a random sample of books downloaded from the Project Gutenberg.

Table 1: Example book corpus

gutenb. id	title	author	gutenberg bookshelf
2095	Clotelle: A Tale of the Southern States	Brown, William	African American Writers
6315	The Awakening of Helena Richie	Deland, Margaret	Bestsellers, American
6971	Judaism	Abrahams, Israel	Judaism
7635	The Disowned — Volume 05	Lytton, Edward and Baron	Historical Fiction
10319	Dave Darrin’s Third Year at Annapolis	Hancock, Harrie Irving	Children’s Book Series

In order to test NLP models, two questions can be examined on the basis of these simple text documents. **Firstly (Q1), does a LDA cluster text documents similar to a human-driven classification?** In this case, this could be validated by the categorization into Gutenberg bookshelves. **And secondly (Q2), what is the best approach to reproduce such a classification using an ANN as a classification model?**
Since very large amounts of data have to be processed and analyzed in order to model the bookshelf classification for a big collection of books, a somewhat reduced approach will be used here. One breaks down a collection of books of different categories into chapters in order to cluster respectively classify these chapters as independent documents. Similarly, you could give an example, assuming N books and separate their chapters and shuffle them, is it possible to use models to reassemble the chapters into stacks that can be assigned to individual books?

The raw text record is now transformed into the tidy format of tidytext (i.e. one word per row). Now it is possible to remove unnecessary stop-words. These words are predefined and can be modified for the appropriate use case if necessary. The words stem from 3 sources, ”onix”, ”SMART” and ”snowball”, whereas the latter two are pulled from the `tm` package [Silge, 2019]. And the Onix stop words are taken from the publicly accessible site lextec.com. An example of stop words can be found in Table 2.

Table 2: Example of stop words from tidytext package

word	lexicon
many	SMART
mrs	onix
was	snowball
sure	onix
you're	snowball
go	onix
c	SMART
where	snowball
then	SMART

Both models that are studied in this thesis, use a bag of words data set as input data. This means a matrix with the documents in the M lines and the frequencies of the V used words in the entries of the columns. The dimension - i.e. the number of words in this "dictionary" V - may be reduced for two reasons. For one thing, a dimension reduction can reduce the fitting time of the model, for another thing, the diversity of the documents can be increased by skilfully reducing certain words, which occur equally frequently in all documents. Now this requires a special measure on which we decide which words to exclude. One can call this reduction of the bag of words dimensionality "embedding". In the course of this work 3 different embedding methods were tested.

1. The reduction by the words that occur with a low frequency.
2. No dimension reduction, i.e. use of the full dictionary of all occurring words.
3. The reduction with the aid of the measure *tf-idf*.

tf-idf is a combination of the term frequency and the inverse document frequency, defined as follows.

$$\textit{tf-idf}(t,d) := tf(t,d) \times \textit{idf}(t)$$

$$tf(t,d) := \frac{f_{t,d}}{\sum_{t_i=1}^{V} f_{t_i,d}}$$

$$\textit{idf}(t) := \ln\left(\frac{M}{n_{d' \in t}}\right)$$

t ... term (word)
d ... document

$f_{t,d}$ ... frequency of term t in document d
M ... number of documents
$n_{d' \in t}$ ... number of documents containing term t

Even if "its theoretical foundations are considered less than firm by information theory experts", *tf-idf* "has proved useful in text mining" [Silge and Robinson, 2017]. In this thesis the term frequency and the *tf-idf* are not to be used directly for the analysis of the texts but mainly for setting up the bag-of-words datasets. It shall be investigated whether the use of different embeddings has an influence on the text analysis itself.

4.1 LDA applied to Textbook Chapters

As LDA is a classification algorithm, the research question Q1 should be addressed first. I.e. to what extent does the clustering of the LDA model correspond to the mapping of chapters to books? In the first attempt only 6 books are sampled. That is, there are 6 categories, because as mentioned in 4 all these 6 books were taken from different bookshelves.

As described in Chapter 3.1, it is possible to calculate the LDA model by means of two different algorithms. These are the VEM algorithm and Gibbs sampling. In this study we examined both methods. Clearly, the algorithms may differ in the speed of the calculation. However, it does not have to be the case that both calculation methods deliver the same results.

Three fundamentally different approaches should be distinguished for embedding:

1. the entire dictionary, i.e. all words occurring in the corpus are used for the bag of words.
2. for the bag of words only words are used which occur at least 2 times in the whole corpus. This reduces the number of used words and therefore the dimension of the data by nearly 50%.
3. the bag of words is reduced by the same amount as in case two, but not according to absolute frequency, but according to *tf-idf*.

These approaches are reviewed with regard to the goodness of the fit and the speed of the computation. One intuitively may assume that the calculation takes longer for data with higher dimensionality.

For a clustering model there are two useful methods to evaluate the fit. On the one hand it is possible to fit the model with training data and to evaluate the goodness of the fit using the test data. On the other hand, it is a comparison of the classification of the model with the categories given by the data, whereby the model does not learn from these categories. The latter evaluation method is clearly less computationally intensive, and the comparison of the entire data set instead of only the test part of the data has a positive effect on the variance. Starting this analysis the second approach is used primarily.

The findings of this study relating to the LDA model are documented in detail. You will find the documentation regarding the LDA model as well as the other models in the appendix. For the sake of clarity, a separate document with the corresponding results was created for each analyzed example and each model respectively. To ensure reproducibility, the entire code is attached to the work by means of these documentations, and will be published on GitHub as well.[3]

The computation of the LDA model via VEM takes a somewhat longer time in the two samples evaluated. In all cases investigated, i.e. across the two samples taken and for both VEM and Gibbs sampling, the calculation for the tf-idf embedding appeared to be the fastest. The calculation time for embedding according to term frequency and full embedding differs less and the calculation of the term frequency embedding is not faster in every case. While a reduction via term frequency only cuts the most rarely used words, a reduction via tf-idf also cuts out frequent words as long as they occur frequently in all documents. Thus the tf-idf reduction has more impact on the distribution within each document compared to the effect of the term frequency reduction, which only affects the tails of the marginal distribution. This supports the assumption that a higher degree of difference in the documents has a positive effect on the calculation speed.

With regard to the quality of the fit, it is apparent that Gibbs sampling performs much better than models fitted by VEM. The best fit using VEM still shows a higher misclassification rate than the poorest fit using Gibbs sampling in the examples examined. Surprisingly, the models with the tf-idf embedding do not perform significantly better than the other approaches, as one might expect that more differentiated word distributions of the documents not only allow a faster calculation, but also result in a higher accuracy. In fact, frequency 2 embedding provides better accuracy results throughout the studied examples.

It was also analyzed how the accuracy will change when using more categories. For this purpose the number of books was increased to 10. The fit via VEM was still much worse than the fit via Gibbs sampling. The model

[3] `https://github.com/SebastianKnigge/Master_Thesis/tree/master/Documentations`

fitted with VEM showed a missclassification rate almost twice as high as the one fitted with Gibbs sampling. For both methods the accuracy deteriorated for more topics used. This sounds intuitive since the clustering algorithm now has to distinguish between more topics. The box plot in Figure 5 illustrates the LDA algorithm in the different use cases. Here we are comparing the investigated cases. Clearly the difference in the misclassification rate between the examples with 10 and 6 categories becomes apparent. It also illustrates how different the two calculation methods - Gibbs sampling and VEM - are.

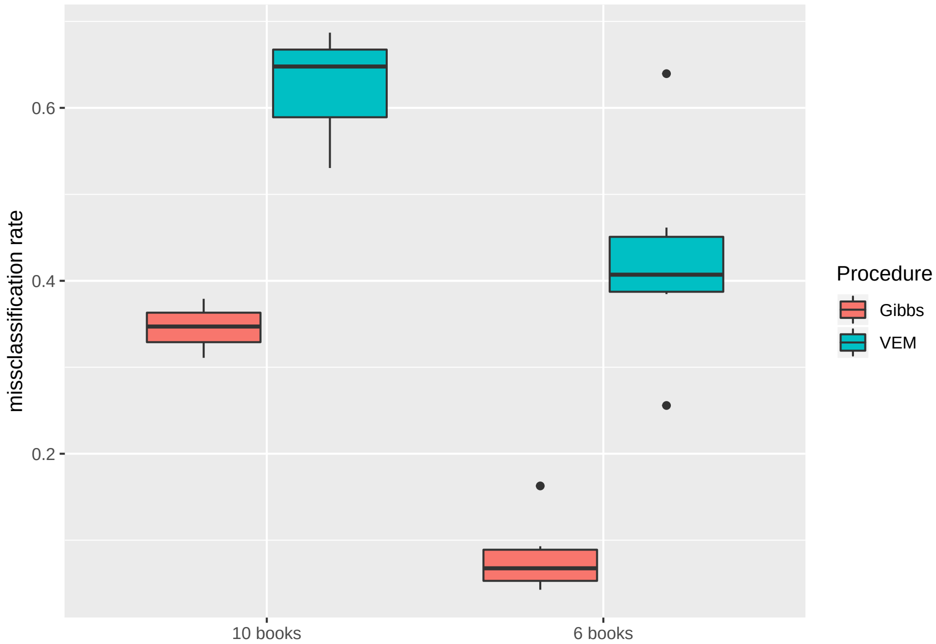

Figure 5: Boxplot of the all examples analyzed. Split for the number of categories (books) as well as for the calculation procedures (VEM algorithm and Gibbs sampling)

The classical method to evaluate the model is to split the data set into a training set and a test set, in order to fit the model on the training data and to evaluate it on the test data. To calculate valid results 59 random clustered samples were used with splits of 90% training sample and 10% test sample applied. The results for those models that were only fitted to a subset of the data are significantly worse than the comparable models that were trained and tested on the entire dataset. However, since here a clustering model is discussed that does not rely on training data, such as a classification model, it is technical correct to evaluate the model using training data.

4.2 ANN applied to Textbook Chapters

In what context is it now possible to apply an artificial neural network to this data? Obviously neural networks are a very versatile tool, but here we focus on the reproduction of a known classification of the documents (chapters) by a suitable model. With the help of such networks in the sense of a classification algorithm, research question 2 will be addressed in this chapter. Again, for the shake of reproducibility the entire code including outputs of the computations will be found in the documentation in the appendix.

Primarily the question shall be addressed whether a neural network is suitable as a classification algorithm for NLP with bag of words data. First of all, the architecture of the network used will be explained (see Figure 6 for comparison). The network essentially consists of three layers. The lowest V

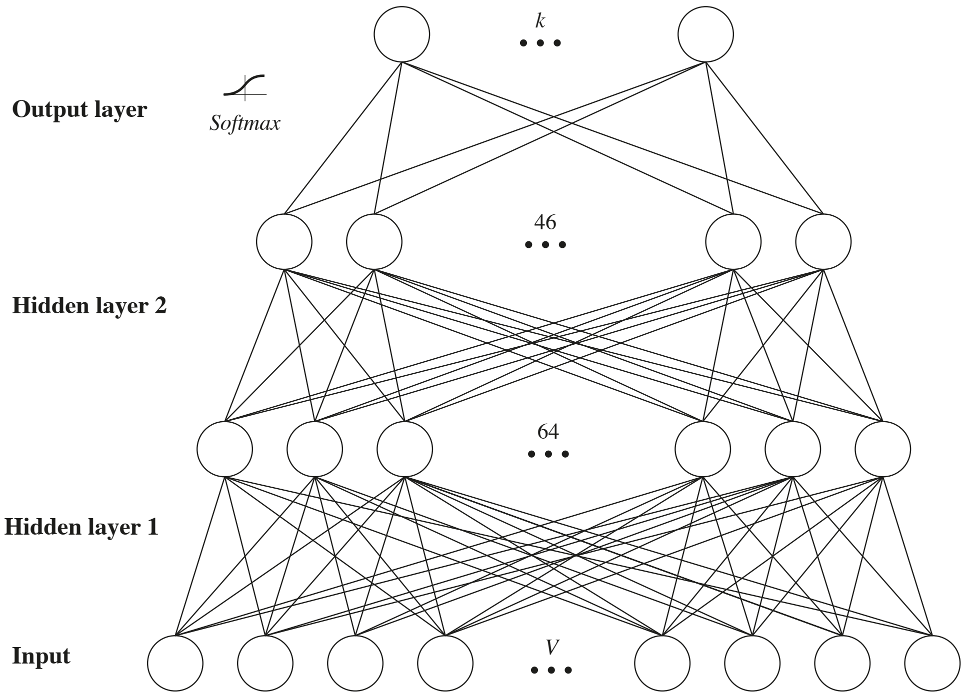

Figure 6: The network contains 3 layers. The input data is the the bag of words, which has dimension V. The first two layers are hidden, containing 64 and 46 neurons respectively. The last layer leads to the k classes (here the number of topics or books). The activation function is softmax.

neurons represent the dimension of the input, i.e. the dimension of the bag of words input vectors rather than an actual layer. The network contains two hidden layers, each containing 64 and 46 neurons. The last output layer contains k neurons, which is the number of topics - in this case books - to classify. Note that the architecture of the network was arbitrarily chosen and that there may be room for improvement.

Whether a network is a classification system or e.g. a regression system depends almost exclusively on the activation function in the last so-called output layer. The "softmax function", also known as the "normalized exponential function" [Bishop, 2006, p. 115], is commonly used for a classification network. The softmax function maps to a $(0, 1)$ space, where the sum of the vector in the image space corresponds to 1. Thus, the output corresponds to a discrete distribution over the different classes. The classification is made using the maximum probability within this distribution per instance.

$$P(y = i|\mathbf{x}) = \frac{e^{\mathbf{x}^t \mathbf{w}_i}}{\sum_{j=1}^{k} e^{\mathbf{x}^t \mathbf{w}_j}} \tag{16}$$

Where $\mathbf{x}$ is the vector of the output of the second hidden layer, and $\mathbf{w}_i$ are the weights of the i-th neuron in the output layer.

Considering the computing power of the used machine and the limited computing time combined with overfitting, a training using 5 epochs and a batch size of 512 turned out to be suitable. In contrast to the LDA model, however, this procedure must be fitted using training data and evaluated using test data. For this purpose, a proportion of 10% test data, 70% training data and 20% validation data for learning has been used.

Considering the computing power of the used machine and the required computing time combined with overfitting, a training using 5 epochs and a batch size of 512 turned out to be suitable. In contrast to the LDA model, however, this procedure must be fitted using training data and evaluated using test data. For this purpose, a proportion of 10% test data, 70% training data and 20% validation data for learning has been used. Note that in a deep learning setting a validation set is essential to prevent overfitting during learning. The test set is then used to evaluate the model via repeated sampling with 59 iterations.

The results of the repeated sampling show that a large part of the classifications by the ANN exactly match the true values of the test data set. In very few samples the classification was incorrect for 10-20% of the test sample. Therefore misclassification rate of the investigated samples was in the range of 1-2% with a very small variance of 0.1-0.2%. These results in this rather small sample support the claim that classification via ANN with bag of words data works very well.

5 Analysis of EUROSTAT Documents

In this application, we would like to focus on the classification of technical, regulatory documents and guidelines. The ESS Vision 2020 ADMIN (Administrative data sources) project aims at "guarantee the quality of the output produced using administrative sources, in particular the comparability of the statistics required for European purposes"[Eurostat, 2019]. This project contains a collection of 28 official Eurostat documents, i.e. guidelines, methodological definitions, and manuals, e.g. on data access. Table 3 displays the full corpus including all documents.

This chapter will examine how the two methodologies investigated so far can be efficiently applied to the type of documents described above, in order to gain informational value. The research questions in this case are slightly modified and aim at a more specific problem solving approach in the context of the evaluated data. **Q3: To what extent can the LDA algorithm support a decision maker in clustering the ESS Vision 2020 ADMIN documents?** Question Q4 builds directly on question Q2, differs however regarding the documents examined. **Q4: How well may the classification of ESS Vision 2020 ADMIN documents be reproduced using an ANN?** In this case, we particularly focus on the setups of the two methods we already optimized for the application to the Gutenberg text data. This means that the same hyper-parameters and fitting methods are used as those tested and proven in chapter 4, because the aim of that example was to find an optimal generic architecture for problems of this kind.

Only a very small amount of preprocessing was necessary for the documents that were taken directly from the website. Solely for a few documents the table of contents had to be removed, as the same table of contents appeared several times for different documents. The same stop words were excluded as in Chapter 4.[4] Consideration must be given to further circumstances of the regulatory documents. Remember that there are many formulas and technical abbreviations in the documents, so each variable, each estimator, and each index is included as a single word in the bag of words. These terms sometimes have a big influence on the documents, because they are very specific for individual documents and occur quite often. To avoid this, all mixed words which include characters and numeric attributes, and all terms with special characters (e.g. Greek letters) are also excluded.

[4] The stop word dictionaries "onix", "SMART" and "snow- ball" are used, as they are provided by the `tm` package [Silge and Robinson, 2017].

Table 3: Entire list of documents

Doc. No.	Document title
1	admin-wp1.1_analysis_legal_institutional_environment_final.pdf
2	admin-wp1.2_good_practices_final.pdf
3	admin-wp2.1_estimation_methods1.pdf
4	admin-wp2.2_estimation_methods2.pdf
5	admin-wp2.3-estimation_methods3.pdf
6	admin-wp2.4_examples.pdf
7	admin-wp2.5_alignment.pdf
8	admin-wp2.5_editing.pdf
9	admin-wp2.5_greg.pdf
10	admin-wp2.5_imputation.pdf
11	admin-wp2.5_macro_integration.pdf
12	admin-wp2.5_macro_integration.pdf
13	admin-wp2.6_good_practices.pdf
14	admin-wp2.6_guidelines.pdf
15	admin-wp3.1_quality1.pdf
16	admin-wp3.2_quality2.pdf
17	admin-wp3.3_quality.pdf
18	admin-wp3.4_quality.pdf
19	admin-wp3.5_quality_measures.pdf
20	admin-wp3_coherence.pdf
21	admin-wp3_growth_rates.pdf
22	admin-wp3_suitability1.pdf
23	admin-wp3_suitability2.pdf
24	admin-wp3_suitability3.pdf
25	admin-wp3_uncertainty.pdf
26	admin-wp5_frames.pdf
27	admin-wp5_frames_examples.pdf
28	admin-wp5_frames_recommendation.pdf

5.1 LDA applied to EUROSTAT Documents

A fundamental problem when clustering text documents is that for certain corpora there may not be a unique grouping of the documents, even if the number of clusters is fixed. Logical, thematic clustering always has a certain uncertainty depending on the aspects according to which it is clustered. For this reason, it is reasonable to compare the statistical clustering to a human-made grouping. This corresponds to the procedure of chapter 4.1, where the clustering of books by the LDA algorithm was compared with the man-made classification of Gutenberg bookshelves (due to the long computing time – in fact - the approach was adopted for comparing chapters with the allocation to books of different bookshelves).

*Note: The terminology used in this chapter refers to the expert assessment when mentioning **groups** and to the clusters of the LDA model when discussing **topics**.*

In this case it is reasonable to consult an expert to get a suitable initial thematic grouping. Univ.-Prof. Dr. Wilfried Grossmann from the Faculty of Computer Science at University of Vienna is an accredited expert in this field and based on his expertise he established a grouping for 7 groups for the documents described here. He proposed the grouping into the following thematic clusters:

1. Legal documents
2. Methods
3. Examples
4. Details
5. Quality
6. Tests
7. Frames

The model already known from the example in chapter 4.1 with the same specifications was used. This means that 7 categories were used, and Gibbs sampling was chosen for the fit, after it turned out that the LDA model clustered better when used with the Gutenberg data. Also the embedding via minimum term frequency 2 was used, since in the course of the work for the LDA model it appeared to be a robust method in terms of accuracy. Once again, for clustering, the comparison to a man-made grouping is not trivial, because the algorithm can recognize patterns other than those assigned by humans in a certain context. Instead of evaluating the fit using a very strict measure, such as accuracy or misclassification ratio, a logical comparison of the two classifications seems to be more appropriate.

In a first step, it makes sense to find out according to which aspects the algorithm has clustered, i.e. which topics are dominant in the found clusters. I am using word clouds to illustrate the contents of the individual clusters. In this case I made use of the package wordcloud which automatically renders wordclouds based on a frequency distribution which is passed to the *wordcloud*() function along with the terms. Analogous to [Winter, 2017] here the *tfidf* measure is used instead of the absolute term frequency. This is advantageous as the wordclouds do not resemble each other so much when deploying *tfidf*, especially since the documents all originate from the underlying topic statistics.

Figure 7: Wordclouds for the clustered topics via LDA – using tfidf word proportions

Some groups can be recognized very well in the clusters detected by the LDA. For example, the group "legal documents" corresponds exactly to one cluster. The group "Frames" can also be identified from the wordclouds as a separate topic. Group 4 is also extracted relatively well, but due to the wordcloud it may rather be viewed as topic "documents on Bayesian statistics", rather than "Details". Other groups such as "Quality" and "Tests" are mixed together and divided into two new clusters. Obviously, the model sticks in this case much more to the individual words than to the latent groups as they were assessed by the Expert.

The descriptive results of this example mainly concern embedding. Due to the number of documents (28 documents), the dimension of the bag of words is smaller. It amounts to almost 8,600 words, for the full embedding compared to 15,000 words for the analogous dictionary of the Gutenberg data. However, the pruning of the bag of words by the frequency 2 embedding is less in comparison. While the dimension of the full bag of words to the frequency 2 bag of words decreased by about 40% for the Gutenberg data, the dimension for the Eurostat data decreased by only 30%. In this application, I chose embedding via term frequency 2. This is - as mentioned above - because it demonstrated to be the best alternative in combination with the LDA model in the previous analysis. For comparison, however, the tfidf embedding with the same dimensions as the *tf-2* bag of words was also tested. Again, the model fitted slightly faster for the *tfidf* embedding compared to the frequency 2 embedding.
Numerical results regarding the fit may also be given, even if they do not guarantee good comparability due to the very specific application. The accuracy ratio using the same calculation as in chapter 4.1, was material better for the frequency 2 embedding than for the *tfidf* embedding (misclassification rates: 0.5 compared to 0.79). Still, these results are way worse than what experienced in the Gutenberg data example. It should be noted that the use case differs considerably from the first example, and that there are considerably fewer documents involved.

A Appendix

References

[A. Gelman, 2014] A. Gelman, J. Carlin, H. S. D. D. A. V. D. R. (2014). *Bayesian Data Analysis*. Chapman and Hall/CRC.

[AP Dempster, 1977] AP Dempster, NM Laird, D. R. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Rojal Statistical Society*.

[Bahjat, 2006] Bahjat, F., e. a. (2006). Multivariate logistic models. *Biometrika Trust*, 93(4):1011–1017.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

[Blei, 2012] Blei, D. (2012). Probabilistic topic models. *Communications of the ACM*, 55(4):77.

[Blei, 2003] Blei, D., N. N. J. M. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.

[Chollet, 2018] Chollet, F., A. W. (2018). *Deep Learning with R*. Manning.

[DeJong, 1979] DeJong, G. (1979). Prediction and substantiation: a new approach to natural language processing. *Cogn. Sci.*, 3(3):251–271.

[Eurostat, 2019] Eurostat (2019). Ess vision 2020 admin (administrative data sources).

[Grossmann, 2004] Grossmann, D., F. O. (2004). *Information Retrival - Algorithms and Heuristics*. Springer, 2 edition.

[Harris, 1951] Harris, Z. (1951). Methods in structural linguistics. *Linguistic Society of America*.

[Hart,] Hart, M. Project gutenberg.

[Hecht-Nielsen, 1988] Hecht-Nielsen, R. (1988). Neurocomputing: picking the human brain. *IEEE Spectrum*, 25(3):36–41.

[Hornik, 2011] Hornik, K., B. G. (2011). topicmodels: An r package for fitting topic models. *Journal of Statistical Software*, 40(13).

[Jacobs, 1993] Jacobs, P., R. L. (1993). Innovations in text interpretation. *Artificial Intelligence*, 63:143–191.

[Jordan, 1999] Jordan, M., e. a. (1999). An introduction to variational methods for graphical models. *Kluwer Academic Publishers - Machine Learning*, 37:183–233.

[M. Steyvers, 2006] M. Steyvers, T. G. (2006). Probabilistic topic models. *Latent Semantic Analysis: A Road to Meaning*.

[Manning, 1999] Manning, C., S. H. (1999). *Foundations of Natural Language Processing*. MIT press.

[Martinez, 2010] Martinez, A. (2010). Natural language processing. *John Wiley and Sons, Inc.*, 2:253–257.

[McCulloch and Pitts, 1943] McCulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*.

[Minsky and Papert, 1969] Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT press.

[Mukherjee, 2019] Mukherjee, S. (2019). Lecture notes for probabilistic machine learning. Duke University.

[Noble, 1988] Noble, H. (1988). *Natural LAnguage Processing*. Blackwell Scientific Publications.

[Powieser, 2012] Powieser, M. (2012). Latent dirichlet allocation in r. Master's thesis, Vienna University of Economics and Business.

[Robinson, 2018] Robinson, D. (2018). *Package 'gutenbergr'*.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron, a probabilistic model for information storage and organization in the brain. *The Psychological Review*.

[Silge and Robinson, 2017] Silge, J. and Robinson, D. (2017). *Text Mining with R: A Tidy Approach*. O'Reilly Media.

[Silge, 2019] Silge, J. e. a. (2019). *Package 'itdytext'*.

[Wainwright and Jordan, 2008] Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305.

[Werbos, 1974] Werbos, P. (1974). *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University.

[Winter, 2017] Winter, K., R.-M. S. G. W. F. I. M. W. (2017). Characterizing regulatory documents and guidelines based on text mining. *Lecture Notes in Computer Science*, 10573.

List of Figures

List of Tables