FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

ASSIGNMENT OF BACHELOR'S THESIS

Title:	Product review sentiment analysis in the Czech language
Student:	Lukáš Langr
Supervisor:	Ing. Daniel Vašata, Ph.D.
Study Programme:	Informatics
Study Branch:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	Until the end of summer semester 2019/20

Instructions

Sentiment analysis is an approach that aims to extract the polarity of a given text. Such polarity may, for example, correspond to a positive or negative review of some product. The aim of this work is to review and apply state of the art methods of sentiment analysis on product reviews in the Czech language.

1) Review and theoretically describe state of the art approaches for sentiment analysis. Focus on the various representations of words/documents like tf-idf or vector representations of words.

2) Use or implement at least two of the reviewed methods and experimentally compare their performance on reviews in the Czech language. Avoid implementing anew those methods that can be easily taken over
from available implementations.

3) Propose a direction for further improvement of selected approaches.

References

Will be provided by the supervisor.

Ing. Karel Klouda, Ph.D.
Head of Department

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague January 30, 2019

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Product review sentiment analysis in the Czech language

Lukáš Langr

Department of Applied Mathematics
Supervisor: Ing. Daniel Vašata, Ph.D.

May 14, 2019

Acknowledgements

I would like to thank my supervisor Ing. Daniel Vašata, Ph.D. for all the thought-provoking consultations we had during the creation of this thesis. It would have been much harder for me to complete this thesis without his guidance.
Another enormous thank you goes to my girlfriend Julie for helping with proofreading and for her support throughout the writing of this thesis.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on May 14, 2019 ……

Czech Technical University in Prague
Faculty of Information Technology

Citation of this thesis

Langr, Lukáš. *Product review sentiment analysis in the Czech language.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

Tato práce poskytuje bližší pohled na současně nejmodernější metody reprezentace dokumentů pro účely analýzy sentimentu. Přestože se mnoho nedávných článků soustředí buď na angličtinu nebo čínštinu, tato práce poskytuje unikátní hodnocení daných metod z pohledu českého jazyka. Převádíme české rezence do různých reprezentací a za pomocí modelů strojového učení na nich provádíme klasifikaci do několika tříd sentimentu. Dosažená přesnost předčila naše očekávání i podobné výzkumné články v českém prostředí používající stejný dataset. Věříme, že tato práce bude základem dalšího rozsáhlejšího výzkumu těchto reprezentací.

Klíčová slova analýza sentimentu, klasifikace, strojové učení, recenze, word2vec, BERT, čeština, zpracování přirozeného textu

Abstract

This thesis provides a closer look at the state of the art methods of representing documents for sentiment analysis tasks. As many of the recent articles only focus on either the English or the Chinese language, this thesis provides a unique evaluation of those methods from the perspective of the Czech language. We use various representations on reviews in the Czech language and perform a multiclass sentiment classification via machine learning models. Our achieved accuracy supersedes expectations and similar research articles using the same dataset in the Czech field. We believe this thesis will be a base upon which more extensive research of the possibilities of these representations will be conducted.

Keywords sentiment analysis, classification, machine learning, reviews, word2vec, BERT, Czech language, natural language processing

Contents

List of Figures

List of Tables

Introduction

The rise of e-shops, social media and enormous amounts of user generated text content in general has made it impossible for a person to read and evaluate their sentimental meaning. Hence the need for a scientific way for determining the polarity of a piece of text was created.

The field of sentiment analysis (also known as opinion mining) has been a trending research subject ever since. It combines the elements of machine learning with regular and computational linguistics to try to understand documents written in natural language and classify them as varying degrees of positivity, negativity and neutrality.

Many companies ranging from technology giants and online retail stores to small restaurants rely on their costumers' feedback to deliver the best services and products. Sentiment analysis allows them to use computers to "read" through any number of costumer reviews and filter out the positive feedback from the negative experiences that could be improved on in the future.

The intention of this thesis is to experiment with different document representations for sentiment analysis done by machine learning.

In the first chapter we are going to explain the origin of sentiment analysis and some necessary theoretical background. Then we are going to present tools and frameworks used for sentiment analysis with machine learning.

Finally, in the experiments part of this thesis, we are going to focus on adapting the state of the art sentiment analysis techniques for the classification of product reviews in the Czech language. Most of the recent research has been done on texts in either the English or the Chinese language. We want to check if or how well can those new technologies be used on Czech texts and potentially give Czech businesses the same tools their English and Chinese counterparts already have.

Goals

Our goal in the theoretical part of this thesis is to research the state of the art methods for sentiment analysis. Especially, our focus will be on the various representations of documents to be used with standard supervised machine learning algorithms.

In the implementation part we are aiming to adapt researched methods for the use on reviews in the Czech language. We are going to score each created model with test data and discuss the results.

Our ultimate goal is to decide whether these models and representations are suitable for use with the Czech language or possibly suggest any improvements.

Chapter 1

Sentiment Analysis

In recent years we have seen a boom in NLP (Natural Language Processing) research. One of the most prominent NLP topics is sentiment analysis. The purpose of a sentiment analysis is to take texts written by people, usually some sort of reviews or opinion posts, and classify them into one of these three categories:

positive a text written by someone who was satisfied with the subject,

negative a text written by someone who was unsatisfied with the subject and

neutral a text written by someone who doesn't express an opinion about the subject.

Sentiment analysis was first derived from linguistics and therefore used its tools such as opinion word lexicons, hand-crafted rules or morphological analysis. Researchers have been using these methods in conjunction with mathematical models to determine semantic orientation of adjectives [1] or opinion words [2].

The technological progress in machine learning methods of the mid 2000s has drawn attention of sentiment analysis researchers. Pang and Lee were the first to introduce pure machine learning approaches in [3] into the field of opinion mining on the IMDB movie review dataset [4]. Before them, every other work contained at least some linguistic prior knowledge. Since Pang and Lee the sentiment analysis research has been split into two branches shown in figure 1.1 lexicon-based and machine learning methods. Our interest lies in the latter so the rest of this thesis is going to be about the machine learning side.

On the machine learning side there are many options how to represent textual data for the models to understand. The tried and tested methods are Bag of words and TF-IDF. New, much more sophisticated methods for translating strings into vectors of number have been discovered. Mainly, the

Word2vec model introduced in 2013 by Mikolov et al. in [5] has been used in many sentiment analysis like [6]. The hottest new technology in the field of representing words is BERT, proposed in [7] in 2018.

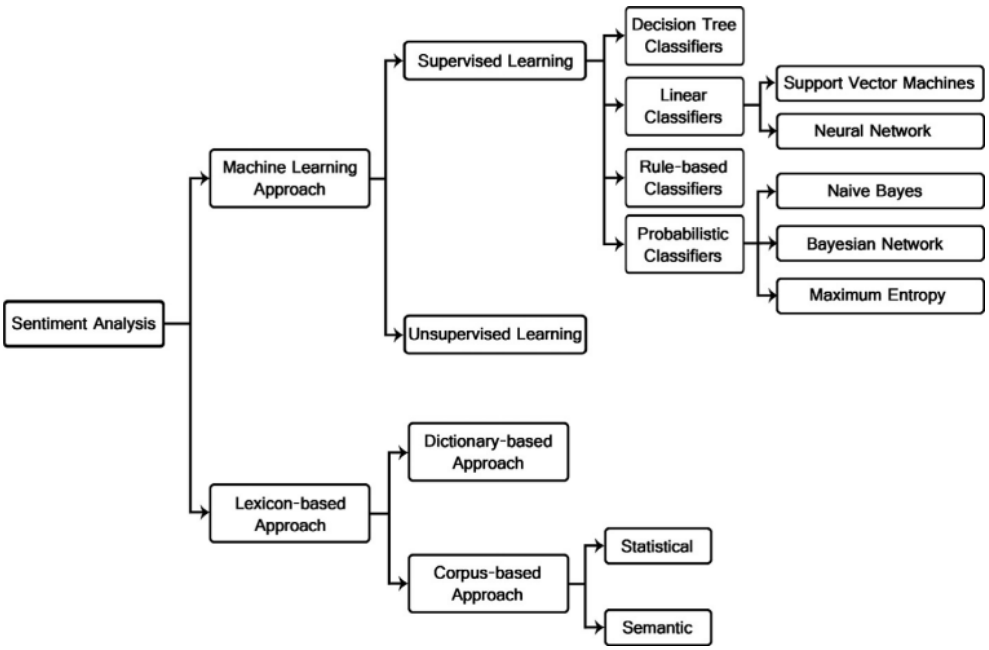

Figure 1.1: Tree of sentiment analysis techniques [8].

1.1 Czech Environment

The first research in the Czech environment was done by Veselovská et al. in [9]. The researchers experimented with annotating text manually and automatically and also built a Naive Bayes classifier trained on the annotated corpus.

In 2012 Steinberger et al. researched a semi-automatic approach for creating sentiment dictionaries in many languages in [10]. They managed to produce gold standard sentiment dictionaries for two languages and translated it automatically into a third using a "triangulation" method.

An in-depth research of machine learning used on Czech social media posts was done by Habernal et al. in [11]. The researches crawled multiple Czech sites and created 3 datasets containing more than 230k of Czech Facebook groups posts, ČSFD[1] movie reviews and Mall.cz product reviews. The Facebook dataset was manually annotated into 3 classes: positive, negative and neutral. Finally, they used these datasets to train the Maximum Entropy

[1] https://www.csfd.cz/

(MaxEnt, Logistic regression) and Naive Bayes classifiers using TF-IDF representation as features.

1.2 Machine Learning Methods

Machine learning (ML) has seen a huge boom in the last decade with the improvements in computational power. That is also why it became a viable strategy for sentiment classification.

In general, machine learning focuses on creating mathematical models and feeding it data for it to learn to recognize patterns. There are two important approaches to machine learning:

Supervised learning model learns from example data with its class indicated

Unsupervised learning model is not given the class of the data, it simply groups similar data together

ML sees sentiment analysis as either binary (positive or negative) or n-ary (varying degrees of positivity, negativity and neutrality) classification problem. Both unsupervised learning for grouping similar texts together and supervised learning for creating classifiers based on annotated inputs are used in the field of opinion mining. With that perspective, we can use our typical supervised classification algorithms to tackle this task.

1.2.1 Classification Workflow

Every classification task follows these steps:

1. Load input data.
2. Split input data into training and testing subsets.
3. Select models and their parameters.
4. Train models using only the training dataset.
5. Evaluate trained models using the testing dataset.

1.2.2 Random Forest

One of the traditionally very well performing ML models is a Random forest classifier. It is an example of an ensemble classification method. First introduced in 2011 in [12], Random forest quickly became a very popular general purpose model.

Random forests are built upon a couple of important techniques, as described below.

Bootstrapping

The bootstrap technique allows us to create multiple data subsets from one dataset by sampling with replacement.

Decision Tree Classifiers

An important role in random forests is played by decision trees. Those are also popular ML classification models. They are binary trees where in each node there is a decision to be made about the input data. If they fulfill a disjunctive condition we move to the left node and if they do not then we move to the right one. Once the input reaches a leaf, it is classified as the class of the majority of the training data that created the leaf node.

Construction of an optimal decision tree is an NP-complete problem. That is why the trees are built using a greedy algorithm C4.5 [13] or C5 with heuristics. The heuristics used is usually information gain measured by quantities like

Entropy $H(\mathcal{D}) = -\sum_{i=0}^{k-1} p_i \log p_i$ or

Gini index $GI(\mathcal{D}) = 1 - \sum_{i=0}^{k-1} p_i^2$,

where there are k values in $\mathcal{D}$ and p_i is the ratio of i-th value in $\mathcal{D}$.

In each step of constructing the tree we want to split the sample data based on the attribute giving us the best information gain (either the highest entropy or gini index). We continue doing this until a stop condition is met. E.g.:

- All the samples belong to the same class.
- None of the remaining features provides any information gain.
- Maximum depth constrain of the tree has been reached.

An example of such a decision tree can be seen at 1.2.

Bootstrap Aggregating (Bagging)

Random forests are produced by bootstrapping a number of random data subsets and then training a small decision tree (a weak learner[2]) on it. When predicting we generate predictions from each decision tree and combine them into one group decision.

The advantage of random forests compared to simple decision trees is a much better bias and over-fitting resistance [12].

[2] A classifier whose accuracy is just above 50 %

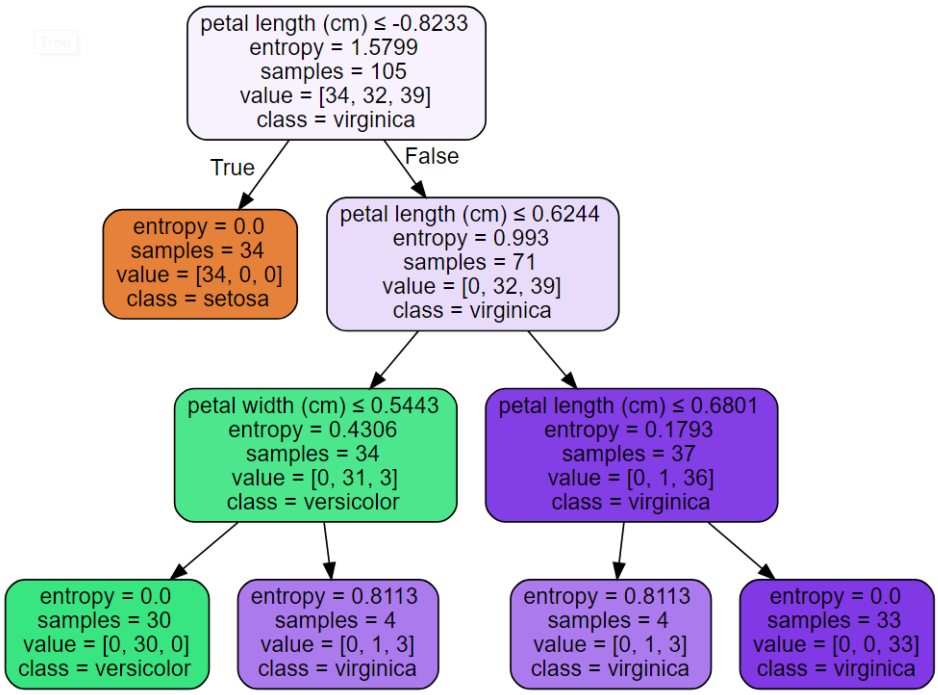

Figure 1.2: Example of a decision tree for classification of the Iris dataset using entropy as an information gain quantity [14].

1.2.3 Logistic Regression

Logistic Regression (also known as Maximum Entropy) is a probabilistic discriminative classification model [15].

When we are trying to predict a variable $Y \in \{0, 1\}$ using logistic regression we change to problem to predicting the probability of

$$\mathrm{P}(Y = 1 \mid X = x) = \sigma(w^T x), \tag{1.1}$$

where X is a feature space and w is a vector of weights. Formula (1.1) takes a linear combination $w_0 + w_1 x_1 + \ldots + w_n x_n$ and returns a probability of the variable $Y = 1$. To keep the result in [0,1] we will use the sigmoid function $\sigma(x) \in [0, 1]$ whose $D_\sigma = \mathbb{R}$. The sigmoid function formula can be seen in (1.2), its derivative in (1.3) and the graph in figure 1.3.

$$\sigma(x) = \frac{\exp(x)}{\exp(x) + 1} = \frac{1}{1 + \exp - x} \tag{1.2}$$

$$\frac{d\sigma}{dx} = \sigma(x)(1 - \sigma(x)) \tag{1.3}$$

The learning of this model is based on the maximum likelihood estimation of the weights with given features. If $p_{Y_i}(x_i, w)$ is a probability of i-th point of

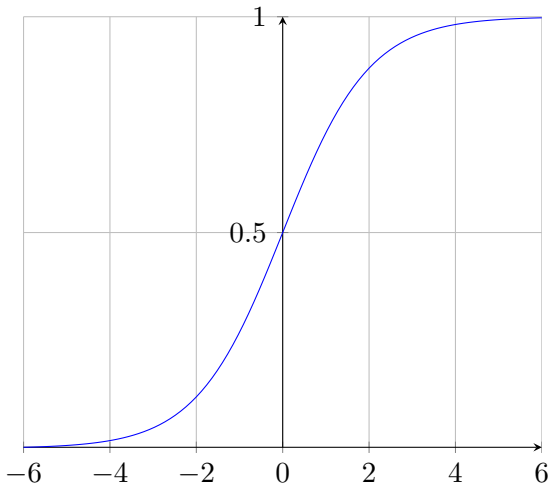

Figure 1.3: Logistic sigmoid function.

the predicted variable with feature values x_i and we assume that all features are independent then the likelihood estimate can be written as follows

$$L(w) = \prod_{i=1}^{N} p_{Y_i}(x_i, w). \tag{1.4}$$

For easier arithmetic manipulation we can maximize a logarithm of L

$$l(w) = \ln L(w) = \sum_{i=1}^{N} p_{Y_i}(x_i, w) \tag{1.5}$$

the gradient of this function can then be written as follows

$$\Delta l(w) = X^\top (Y - P), \tag{1.6}$$

where $P = (p_1(x_1, w) \dots p_N(x_N, w))^\top$. In theory we should be able to find the maximum by solving

$$\Delta l(w) = X^\top (Y - P) = 0. \tag{1.7}$$

Unfortunately this equation does not have an explicit solution. We have to use approximative methods like the Newton method or gradient ascent. [15]

Logistic regression is primarily a binary classification method. To use it in the multiclass scenario we will have to adjust it using a the *one-vs-rest* approach. We train k models for each class and each model is trying to learn if the input is the k-th class of **not**.

1.2.4 Support Vector Machines

Another widely used model for text classification is support vector machines (SVM). It can be a linear or a kernel function classifier both of which are effective and can achieve good performance [15].

SVM basically aims to construct a hyperplane or a set of hyperplanes to separate data into distinct groups, as can be seen in figure 1.4. The larger the distance between the hyperplane and the nearest point in space the better the separation.

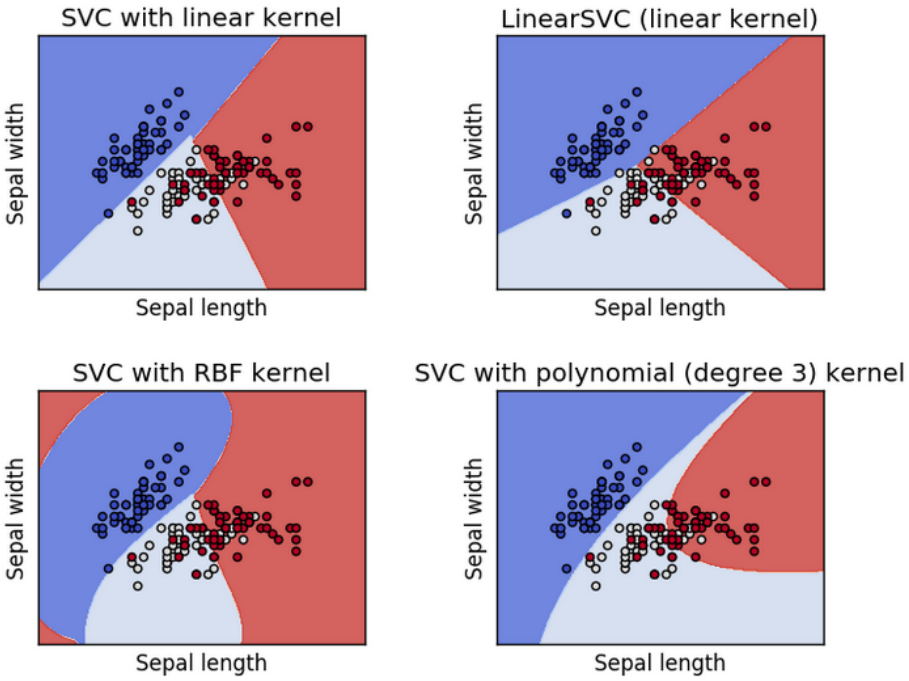

Figure 1.4: Examples of linear and kernel function SVM separation on 2D data [16].

We have a set of $(x_1, y_1) \dots (x_n, y_n)$ where $y_i \in \{-1, 1\}$ indicates the class where x_i belongs. Any hyperplane separating both groups can be written as

$$w \cdot x - b = 0 \tag{1.8}$$

where w is the normal vector to the hyperplane.

Let us assume that the training data is linearly separable. Then

$$w \cdot x - b = 1 \quad \text{and} \quad w \cdot x - b = -1 \tag{1.9}$$

are hyperplanes bounding the region of the margin. The margin is therefore $\frac{2}{\|w\|}$ wide. So to maximize the margin we have to minimize the $\|w\|$.

$$y_i(w \cdot x_i - b) \geq 1 \text{ for all } 1 \leq i \leq n, \tag{1.10}$$

the w and b which solve this problem give us the classifier

$$x \mapsto \operatorname{sgn}(w \cdot x - b). \tag{1.11}$$

For non-separable data we would have to use a hinge loss function.

SVM can use a kernel function to map high-dimensional vectors from the feature space into another space where they are easily comparable. This approach is used for nonlinear classification.

SVMs work with many common kernel functions such as *linear in* (1.12)*, polynomial, radial basis function (RBF) in* (1.13) *and sigmoid.*

$$x \cdot x' = \langle x, x' \rangle \tag{1.12}$$

$$\operatorname{rbf}(x, x') = \exp(-\gamma \|x - x'\|) \tag{1.13}$$

where $\gamma > 0$ specifically $\gamma = \frac{1}{N}$ where N is the number of features.

In a multiclass case we have to train $k(k-1)/2$ different binary SVMs on all possible pairs of k classes. Then we classify test points according to which class has the highest number of "votes". This approach is called *one-vs-one*. It is very computationally intensive and it can also lead to ambiguities in term of classifying one sample into multiple classes. [15]

1.3 Data Representation

Now that we have got our models, the other important problem in machine learning is to choose how the input data will be represented. In sentiment analysis the data are text documents which are somewhat complicated to represent. We want the representation to be vectors of the same length and those vectors should be made of features which should be able to represent all documents.

We are going to use three different representations in this thesis to determine pros and cons of each approach.

1.3.1 TF-IDF

TF-IDF stands for Term Frequency – Inverse Document Frequency. It is a greatly used technique for transforming a set of documents, also called corpus, to a set of vectors of numbers representing said documents. The TF-IDF values are products of two quantities.

TF

The first is term frequency (tf). It measures how much is a word used in a document. There are many ways how tf can be produced. The most common formulas are:

$$\mathrm{tf}(w,d) = \begin{cases} 1 & \text{if } w \in d, \\ 0 & \text{otherwise,} \end{cases}$$

$$\mathrm{tf}(w,d) = \mathrm{f}_{w,d},$$

where $\mathrm{f}_{w,d}$ is the number of occurences of w in d,

$$\mathrm{tf}(w,d) = \log\left(1 + \mathrm{f}_{w,d}\right),$$

$$\mathrm{tf}(w,d) = \frac{\mathrm{f}_{w,d}}{\sum_{w' \in \mathcal{D}} \mathrm{f}_{w',d}}.$$

IDF

The second is inverse document frequency (idf) which quantifies how common or rare a word is in the whole corpus. Its values can be calculated like this:

$$\mathrm{idf}(w,\mathcal{D}) = \frac{|\mathcal{D}|}{|\{d \in \mathcal{D} : w \in d\}|}$$

$$\mathrm{idf}(w,\mathcal{D}) = \frac{|\mathcal{D}|}{1 + |\{d \in \mathcal{D} : w \in d\}|}.$$

The final TF-IDF value for a word w and document $d \in \mathcal{D}$ is a product of term frequency and inverse document frequency [17]

$$\text{tf-idf}(w,d) = \mathrm{tf}(w,d) \cdot \mathrm{idf}(w,\mathcal{D}). \tag{1.14}$$

1.3.2 Word2vec

In 2013, Mikolov at el. in [5] introduced a new way of representing words in computers. They created a two-layer neural network (NN) which takes text as input and produces n-dimensional vectors called word embeddings. What they discovered is that the neural net preserves syntactic and semantic word similarities without requiring labeled data as input (it is unsupervised). E.g. if high dimensional vectors are trained on a large amount of data, the factual relation between two words like Berlin is a capital city of Germany can be applied similarly to France just by using vector arithmetic

$$\mathrm{vec}(\text{“Berlin”}) - \mathrm{vec}(\text{“Germany”}) + \mathrm{vec}(\text{“France”}) = \mathrm{vec}(\text{“Paris”}).$$

Word2vec can work in two different modes. CBOW – continuous bag of words is method when the NN is trying to predict the target word from context and Skip-gram when it is trying to predict the context from the target word. Both architectures can be seen in 1.5.

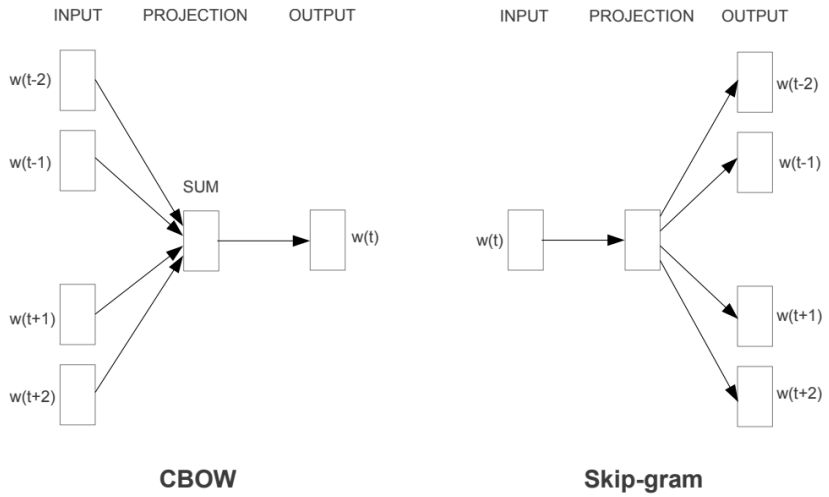

Figure 1.5: Two model architectures of Word2vec [5].

Learning

Word2vec is a fully connected feed forward neural network with a single hidden layer. At the beginning all the words in the training dictionary are represented as one-hot encoded vectors of size V which is the number of unique words in the dictionary. Word2vec takes a sum of s one-hot encoded vectors c from the context of the target word w_t as input, meaning $c = w_{t-\frac{s}{2}} + \ldots w_{t-1} + w_{t+1} + \ldots + w_{t+\frac{s}{2}}$ (assuming s is even for simplicity), and tries to predict w_t.

The prediction is given by terms of a softmax function such as

$$\mathrm{P}(w_t \mid c) = \frac{\exp(\mathrm{score}(w_t, c))}{\sum_{w \in V} \exp(\mathrm{score}(w, c)))} \tag{1.15}$$

where score computes the compatibility of word w_t with the context c. The score function is commonly a dot product. The training is done by maximizing the log-likelihood on the training set. The argument of the maxima of the objective function is the prediction for w_t. This make Word2vec a properly normalized probabilistic model for language modeling. [18]

Negative sampling

Unfortunately the maximum likelihood approach requires calculating the derivative of the sum in (1.15) which takes a lot of computational time even for dictionaries containing tens of thousands of words. Mikolov introduced a method called negative sampling to deal with issue and make training faster. Basically the softmax output layer is replaced by a binary classifier which predicts if w_t belongs between words from its context or not.

At the beginning, we either put the target word w_t with its real context as an input to the classifier and a label of 1, so it learns that w_t belongs together with its context. Or we take w_t and draw some random words from the vocabulary and give to the classifier with a label of 0. This trains the classifier to recognize words that occur together. We can use logistic regression for that while getting rid of the summation from (1.15) which greatly improves training performance.

Thanks to its simplicity and negative sampling Word2vec able to train high quality word vectors really quickly from huge datasets even with one trillion words [5].

1.3.3 BERT

BERT stands for Bidirectional Encoder Representations from Transformers. It is a pre-trained neural network supporting more than 100 languages. BERT was designed for fine-tuning, adding a custom output layer to the pre-trained network. *"…the pre-trained BERT representations can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications"* [7].

BERT outperforms previous methods because it is the first unsupervised, deeply bidirectional system for pre-training NLP. Unsupervised means that BERT was trained using only a plain text corpus which is important because an enormous amount of plain text data is publicly available on the web in many languages. [19]

Transformer

BERT is based upon a Transformer – an attention mechanism which learns contextual relations between words in a text. Transformer does not use a recurrent or convolutional neural net. Instead it uses a so called sequence-to-sequence architecture. Sequence-to-sequence is a neural net that transforms a given sequence of elements, such as the sequence of words in a sentence, into another sequence. This architecture consists of an Encoder and Decoder which can be seen in figure 1.6. The Encoder takes the input sequence and maps it into a higher dimensional space (n-dimensional vector). The same vector is then fed in the Decoder which produces an output sequence. [20, 21]

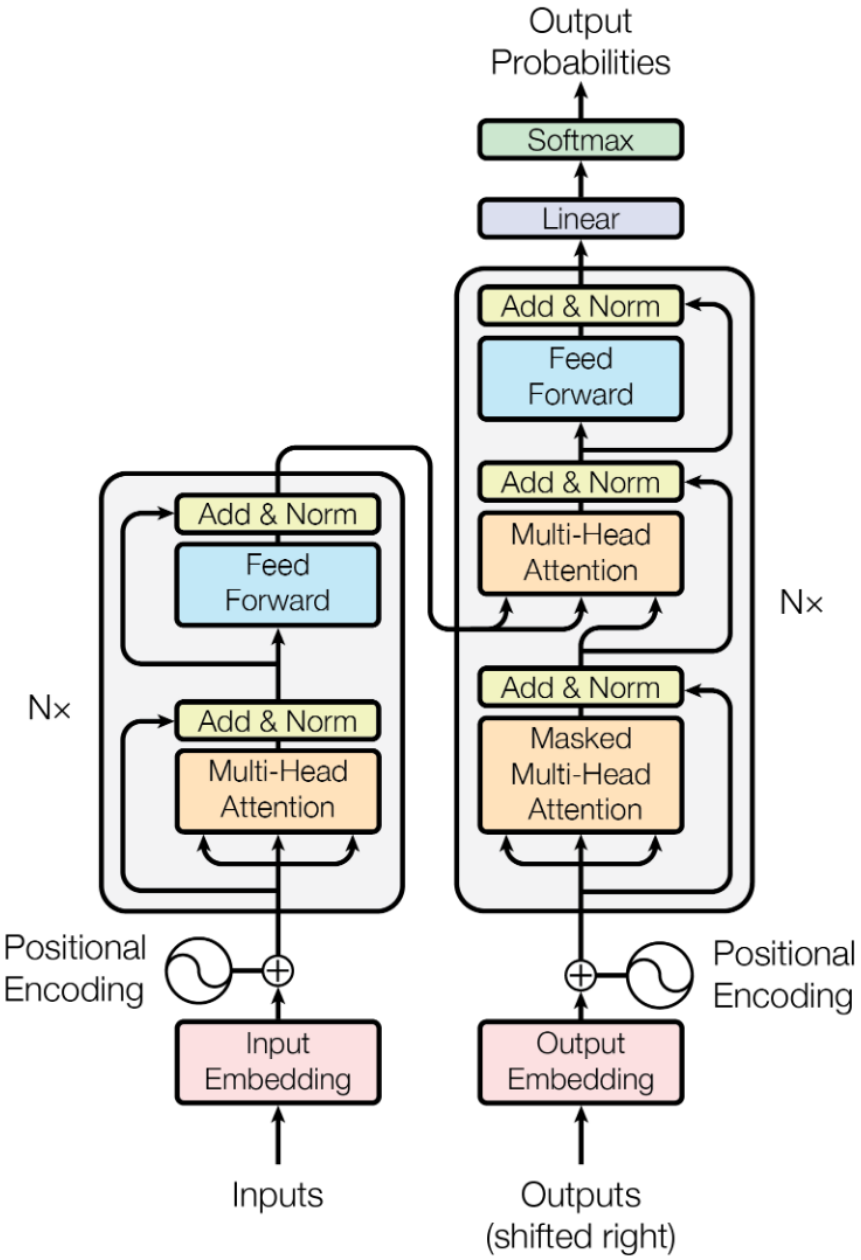

Figure 1.6: The Transformer model architecture. Encoder on the left, Decoder on the right [20].

Attention

"An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all

vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key", [20].

Representation

The process of creating word embeddings in BERT works as follows:

1. BERT represents each token as a embedded vector of selected size n.
2. Then, it adds positional encoding to each token.
3. After that, the data goes through N Encoder blocks.

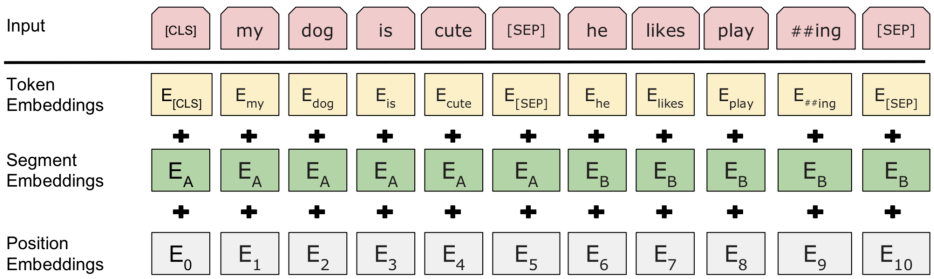

Figure 1.7: BERT input representation [7].

When pre-training BERT, a simple approach is used: they mask out 15 % of the words in the input, run the entire sequence through a deep bidirectional Transformer encoder, and then predict only the masked words. For example:

```
Input: the man went to the [MASK1]. he bought a [MASK2] of milk.
Labels: [MASK1] = store; [MASK2] = gallon
```

In order to learn relationships between sentences, BERT is also trained on a simple task which can be generated from any monolingual corpus: Given two sentences A and B, is B the actual next sentence that comes after A, or just a random sentence from the corpus? [19]

```
Sentence A: the man went to the store.
Sentence B: he bought a gallon of milk.
Label: IsNextSentence

Sentence A: the man went to the store.
Sentence B: penguins are flightless.
Label: NotNextSentence
```

The Transformer uses Multi-Head Attention, which means it computes attention h different times with different weight matrices and then concatenates the results together. For more details see [22].

1.4 Model Evaluation

1.4.1 Classification Accuracy

The accuracy of a classification model can be simply calculated as follows

$$\text{accuracy}(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N} 1(\hat{y}_i = y_i), \tag{1.16}$$

where y is a set of test samples, $\hat{y}$ a set of predictions and $1(x)$ an indicator function which is equal to 1 only if x is true. Clearly $\text{accuracy}(y, \hat{y}), \rightarrow [0, 1]$ where if it is 0 it means that no predictions were correct and if it's 1 then all predictions were correct.

1.4.2 Confusion Matrix

In a multiclass classification problem, there is a need for a technique which measures how many samples of one class have been predicted as some other class. That is precisely what a confusion matrix does. It is defined as a matrix C whose $C_{i,j}$ is equal to the number of observations known to be in group i but predicted to be in group j. In other words the correct prediction are on the diagonal, the rest are misclassifications.

A normalized confusion matrix is defined similarly to the regular one just the $C_{i,j}$ is divided by size of the group i.

$$N_{i,j} = \frac{C_{i,j}}{\sum_{k=1}^{n} C_{i,k}}, \tag{1.17}$$

where n is the number of classes.

An example of a confusion matrix can be seen in figure 1.8. It shows that Class A has 10 samples all of which were predicted correctly to be in A, Class B has 20 sample in total, 15 of which were predicted correctly, 3 were predicted to be in Class A and 2 in Class C. Class C in the bottom row contains 15 samples, 12 were classified correctly and 3 were mistaken for being in Class B. A normalized version of the same matrix as in the example above can be seen in figure 1.9.

1.4.3 Precision

In a simple binary classification, the precision metric is defined as follows

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{1.18}$$

Reality \ Prediction	Class A	Class B	Class C
Class A	10	0	0
Class B	3	15	2
Class C	0	3	12

Figure 1.8: Example of a 3x3 confusion matrix.

Reality \ Prediction	Class A	Class B	Class C
Class A	1	0	0
Class B	0.15	0.75	0.1
Class C	0	0.2	0.8

Figure 1.9: Example of a normalized 3x3 confusion matrix.

where TP is the number of true positives and FP is the number of false positives.

A generalized version of precision for multiclass classification can be calculated as follows

$$\text{Precision} = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \, R(y_l, \hat{y}_l), \tag{1.19}$$

where L is the set of labels and $R(A, B) := \frac{|A \cap B|}{|A|}$.

1.4.4 Recall

In a simple binary classification, the recall metric is calculated as follows

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{1.20}$$

where TP is the number of true positives and FN is the number of false negatives.

A generalized version of recall for multiclass classification can be calculated as follows

$$\text{Recall} = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \, R(\hat{y}_l, y_l), \tag{1.21}$$

where L is the set of labels and $R(A, B) := \frac{|A \cap B|}{|A|}$.

1.4.5 F1 Score

The F-measure can be interpreted as a weighted harmonic mean of the precision and recall. A measure reaches its best value at 1 and its worst score at 0. In F1 score both recall and the precision are equally important. It is calculated as follows

$$\text{f}_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{1.22}$$

A generalized version of the F1 score for multiclass classification can be calculated as follows

$$\text{F}_1 = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \sum_{l \in L} |\hat{y}_l| \, \text{f}_1(y_l, \hat{y}_l), \tag{1.23}$$

where $\text{f}_1(A, B)$ is the binary formula from (1.22) applied for one class from L.

1.4.6 Matthews correlation coefficient

The Matthews correlation coefficient is used in machine learning as a measure of the quality of binary (two-class) classifications. It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes. The MCC is in essence a correlation coefficient value between -1 and +1. A coefficient of +1 represents a perfect prediction, 0 an average random prediction and -1 an inverse prediction. The statistic is also known as the phi coefficient.

$$MCC = \frac{\mathrm{TP} \cdot \mathrm{TN} - \mathrm{FP} \cdot \mathrm{FN}}{\sqrt{(\mathrm{TP} + \mathrm{FP})(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}}. \tag{1.24}$$

In the multiclass case, the Matthews correlation coefficient can be defined in terms of a confusion matrix C for K classes

$$MCC = \frac{c \cdot s - \sum_{k \in K} p_k \cdot t_k}{\sqrt{(s^2 - \sum_{k \in K} p_k^2) \cdot (s^2 - \sum_{k \in K} t_k^2)}}, \tag{1.25}$$

where t_k is the number of times class k truly occurred, p_k the number of times class k was predicted, c the total number of samples correctly predicted, s the total number of samples.

Chapter 2

Tools

2.1 Python

Python[3] is a general purpose programming language created by Guido van Rossum in 1991. It has grown very popular among data scientist and researchers in general for its low barrier of entry and easy syntax. The most important selling feature of Python is its package creating community.

A package is a self-contained code library dealing with a specified task, e.g. working with tables or machine learning algorithms. As the number one choice for data scientists, there are many packages solving everyday tasks in machine learning research while abstracting complicated implementation away. We are going to present a few of those which we used while implementing the tasks of this thesis.

2.1.1 Pandas

In machine learning tasks most of the data in a form of a table. Pandas[4] (derived from an econometric term **pan**el **da**ta) is an easy-to-use framework for selecting data from tables and transforming tables. Therefore, it is a must have tool in any data science related task.

2.1.2 Scikit-learn

The so called gold standard in machine learning libraries for Python is scikit-learn. It offers powerful easy-to-use interface for classification, regression, model selection, preprocessing and basically everything you need to create and evaluate your machine learning models. [16]

[3] https://www.python.org/
[4] https://pandas.pydata.org

It is mostly written in Python but some core algorithms are written Cython[5] to achieve better performance. [16]

The data structures rely on the Numpy package making it compatible with other scientific Python libraries. It also utilizes the Scipy package for efficient algorithms for linear algebra, sparse matrix representation, special functions and basic statistical functions. Scipy has bindings for many Fortran-based standard numerical packages, such as LAPACK. This is important for ease of installation and portability, as providing libraries around Fortran code can prove challenging on various platforms. [16]

We used scikit-learn for all the machine learning models, TF-IDF vectorizer and their evaluation in this thesis.

2.1.3 Gensim

We also used a library for NLP tasks called Gensim (**gen**erate **sim**ilar) created and maintained by Czech programmer Radim Řehůřek [24]. It is an open-source library which has been used in various environments ranging from Amazon to medical companies [25].

Gensim includes streamed parallelized implementations of fastText, word2vec and doc2vec algorithms, as well as latent semantic analysis, non-negative matrix factorization, latent Dirichlet allocation, TF-IDF and random projections.

2.1.4 TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow is built hierarchically as can be seen in figure 2.1. Its lowest layers are CPU, GPU or TPU kernels making it versatile to work on any possible system. The core functions are written in C++ wrapped in a Python interface. It features many libraries for building custom machine learning models as well as testing them and evaluating them. TensorFlow mostly focuses on the implementation of neural networks.

We used it together with BERT to fine-tune it for sentiment analysis purposes.

[5] Cython is a programming language that makes writing C extensions for the Python language as easy as Python itself. It aims to become a superset of the Python language which gives it high-level, object-oriented, functional, and dynamic programming. [23]

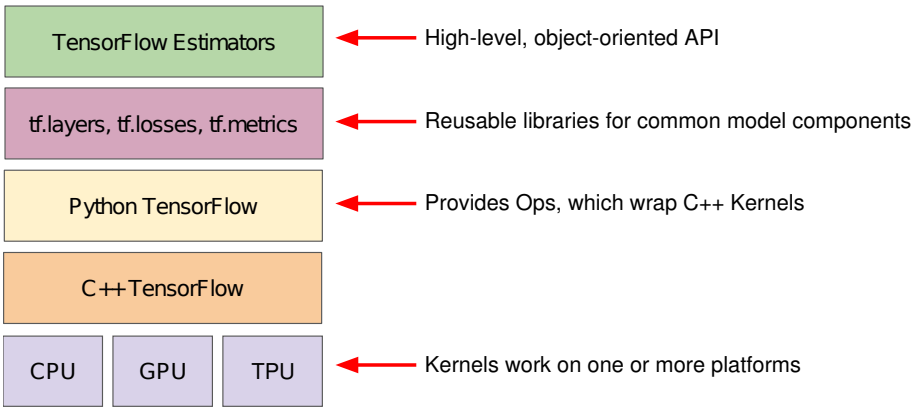

Figure 2.1: TensorFlow toolkit hierarchy.

2.1.5 BERT

The developers of BERT offer a total of seven pre-trained models for download. Two of the seven are multilingual models supporting 104 languages including Czech. I chose the newest Cased (supports lowercase, uppercase and accented letters) model with 12 layers, hidden size of 768 and 12 attention heads. [19]

The data was pre-trained on the top 100 languages with the largest Wikipedias. The entire Wikipedia dump for each language (excluding user and talk pages) was taken as the training data for each language. However some of the sizes of the Wikipedias were smaller than others, therefore those languages are not represented as much as the others. [19]

2.2 Jupyter and Google Colaboratory

"The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text." [26] It is used in scientific environments exactly for above mentioned features. Jupyter notebooks (formerly known as IPython) support Markdown for explanatory text, LaTeX for mathematical equations and a large amount of popular programming languages for interactive applications, e.g.:

- Python
- Julia
- R
- Scala

Google Colaboratory (Colab) is a notebook environment with similar features to Jupyter. Additionally, Colab is completely cloud based and offers

powerful hardware support for faster computation. That is very useful in the NLP field since the most of its methods are computationally demanding.

Chapter 3

Experiments

3.1 Data

Let us start with a description of all the data which we have used during the experiments part of this thesis.

3.1.1 Mall.cz dataset

All the experiments for this thesis have been performed on a corpus consisting of product reviews from a Czech e-shop Mall.cz[6]. This corpus was created by University of West Bohemia in Plzeň as a base for their own sentiment analysis research [11].

There are 145 376 user reviews in the Mall.cz dataset, 103 033 of which are annotated as positive, 31 953 as neutral and 10 390 as negative. Habernal et. al. proposed in [11] that the 4-star user reviews on Mall.cz's represented the neutral sentiment and thus they assigned the 5-star reviews to be positive, 4-star neutral and 3 or less stars to be negative.

Review	**Sentiment**
Nejlepší, nejlehčí, perfektně hrající, dobrá a tvrdá odezva tlačítek.	Positive
Celkem spokojenost, i když stabilita není zas až tak úžasná.	Neutral
Nesplnil očekávání, vrátila jsem. Nekupujte. Nejlevnější není nejlepší.	Negative

Table 3.1: Examples of user reviews from Mall.cz.

[6] http://www.mall.cz/

3.1.2 Wikipedia dataset

To train the Word2vec model we needed an extensive coherent corpus. A great alternative to those license burdened corpora such as Český národní korpus is an absolutely free Wikipedia dump.

We downloaded the complete database of Czech Wikipedia off of their website[7] in a compressed form. At the time of this writing the Czech Wikipedia consisted of more than 420 000 articles [27].

The articles are in an XML structure so we had to strip the tags away to get the raw text. The raw text then had to be decoded from Unicode to ASCII using the Unidecode [28] package, stripped off punctuation and switched to lowercase. It is not absolutely necessary to do the previous steps but it is a good practice which often yields better results.

3.2 Preprocessing

In any kind of NLP task there is a need for preprocessing of the input data. Sentiment analysis is no different especially in the Czech language.

First of all we had to convert all Unicode accented characters to simple ASCII ones. Secondly all the punctuation characters had to be removed and the text was changed to lowercase. Finally the documents were tokenized – converted from strings to arrays of words.

Then we removed *stopwords* (words which are used in almost all texts such as prepositions, conjunctions and all the forms of the verbs *to be* and *to have*) for the TF-IDF and Word2vec models since those words are in almost all the reviews and therefore do not possess any meaning.

There are also advanced methods of preprocessing such as stemming and lemmatization.

Stemming a process of reducing a word to its base form – a stem (not a morfological root) by removing e.g. the word *preprocessing* would be stemmed to *preprocess*.

Lemmatisation a process of grouping together the inflected forms of a word so they can be analysed as a single item called lemma e.g. the word *good* and *good* are both reduced to the same lemma – *good*.

However we chose not to use those because they require prior linguistic knowledge in a form of dictionaries and rules which goes against this thesis being purely about the machine learning approaches. Also, both of these methods were used on the same dataset in [11] therefore we can later compare both approaches to see if they improve the results.

[7] https://dumps.wikimedia.org/cswiki/latest/cswiki-latest-pages-articles-multistream.xml.bz2

3.3 Methodology

All of the following experiments have been performed on the same exact data. The Mall.cz dataset has been split as follows – 75 % training data and 25 % testing data. The absolute numbers can be seen in table 3.2. There is greater number of positive reviews compared to neutral and negative making the dataset unbalanced.

Dataset subset	**Percentage**	**Count**
Training data	75 %	109 032
Testing data	25 %	36 344
Positive training data	71 %	77 261
Neutral training data	22 %	23 936
Negative training data	7 %	7 835
Positive testing data	71 %	25 772
Neutral testing data	22 %	8 017
Negative testing data	7 %	2 555

Table 3.2: Examples of user reviews from Mall.cz.

The evaluation was done by several metrics described in detail in section 1.4. Classification accuracy is a standard metric used in almost all situations. We also chose to include precision, recall, F1 score and the Matthews correlation coefficient to deal with uneven class distribution of the input data.

3.3.1 Binary and multiclass classification

There are two basic approaches to the classification of sentiment. As was said in the introduction to this chapter there are three main classes of sentiment – positive, negative and neutral. We can either make it a binary problem by learning the models only to distinguish between positive and negative sentiments or we can split the sentiment spectrum into multiple classes including the neutral one. E.g. splitting reviews into five categories based on the five stars given by users.

We have decided to perform and compare both binary and a multiclass (including the neutral class, 3 classes in total) sentiment classification of the reviews. Binary classification has been a staple in sentiment analysis ever since [3]. Most research papers completely ignore the existence of a neutral class, a class of documents which do not express positive nor negative sentiment of the author.

According to [29], this attitude towards sentiment analysis is wrong. Of course neutral documents exist very often in the real world and are a very important benchmark for a potential classification model. Therefore they should not be ignored. We will experiment with both approaches to be able to evaluate their pros and cons and compare their performance.

3.4 The TF-IDF scenario

The first of our experiments was a standard TF-IDF representation used with Random Forest, Logistic Regression (Log Reg) and linear SVM classifiers (LSVM). The TF-IDF model removed very common words by filtering out all those which happen to be in more than 10 % of the documents in the corpus to further reduce to vocabulary size. That simultaneously makes the model perform better and more efficiently.

Finally we used the same training data to train the TF-IDF vectorizer and all three ML models. The testing data was transformed to the TF-IDF form and used to evaluate each model.

3.4.1 Results

The binary classification performed really well, reaching around 95 % in almost all the metrics used for evaluation as you can see in table 3.3. The most accurate model for this task turned out to be linear SVM with **96 %** accuracy and **0.73** Matthews correlation coefficient (MCC).

Model	Accuracy	Precision	Recall	F1	MCC
Random Forest	95 %	95 %	95 %	95 %	0.67
Log Reg	94 %	94 %	94 %	93 %	0.58
LSVM	**96 %**	**96 %**	**96 %**	**96 %**	**0.73**

Table 3.3: Accuracy of the TF-IDF based classifiers on a binary problem. Bold numbers denote best results in each metric.

In the multiclass, scenario the most successful model was the Random forest with **84 %** accuracy and **0.60** MCC followed by linear SVM as can be seen in table 3.4. Logistic regression did not perform as well as the other two models, reaching accuracy of 79 % and MCC of 0.48.

Model	Accuracy	Precision	Recall	F1	MCC
Random Forest	**84 %**	**83 %**	**83 %**	**82 %**	**0.60**
Log Reg	79 %	78 %	79 %	77 %	0.48
LSVM	81 %	80 %	81 %	80 %	0.54

Table 3.4: Accuracy of the TF-IDF based classifiers on a multiclass problem. Bold numbers denote best results in each metric.

We can see in the confusion matrix[8] 3.1 that the Random forest hardly misclassified any positive reviews for negative ones. On the other hand the number of neutral reviews classified as positive is quite high.

[8]Neg = number of negatives, Neu = number of neutrals, Pos = number of positives

Reality \ Prediction	Neg	Neu	Pos
Neg	1243	437	875
Neu	105	4205	3707
Pos	59	807	24906

Figure 3.1: Confusion matrix for the Random forest multiclass classifier using the TF-IDF representation.

Logistic regression mistook more neutral reviews for positive ones than it correctly classified in figure 3.2. SVM also did not achieve good results when distinguishing between positive and neutral as can be seen in figure 3.3.

3.5 The Word2vec scenario

Our second experiment was based on the idea that Word2vec vectors keep semantic similarity. We could then transform all the words in a document to Word2vec representation and average them to get a single vector for representing the whole document. We have been inspired for this approach by [30]. This vector carries information about all the semantic meaning of each word and therefore also their sentiment value.

First of all, we had to pre-train Word2vec on a large meaningful corpus. We used the Wikipedia dump concatenated with training part of the Mall.cz dataset. We chose 300 feature vectors and a 10 word context window as hyperparameters of the Word2vec model. The Word2vec model was trained in the CBOW mode because it is faster and it has better representations for more frequent words.

The transformed data was then used to train the Random forest, Logistic regression and linear SVM classifiers. Similarly, transformed testing data was then used to evaluate the performance.

Reality \ Prediction	Neg	Neu	Pos
Neg	1081	678	796
Neu	190	3201	4626
Pos	74	1172	24526

Figure 3.2: Confusion matrix for the Logistic Regression multiclass classifier using the TF-IDF representation.

Reality \ Prediction	Neg	Neu	Pos
Neg	1389	567	599
Neu	280	3765	3972
Pos	131	1295	24346

Figure 3.3: Confusion matrix for the Linear SVM multiclass classifier using the TF-IDF representation.

3.5.1 Examples

With the Word2vec pre-trained on Czech Wikipedia, we tried some its advertised capabilities like finding similar words, deciding which word does not match the others and the vector semantic arithmetic.

In the first case, it correctly predicted that the string "cvut" (ČVUT) belongs together with other Czech technical universities and words like *infomatics* and *engineering*.

```
model.wv.most_similar("cvut")
```

```
[('vut', 0.878714382648468),
 ('vscht', 0.8565627932548523),
 ('zcu', 0.8051557540893555),
 ('elektrotechniky', 0.788162350654602),
 ('informatiky', 0.7844816446304321),
 ('chemicko', 0.7706067562103271),
 ('inzenyrstvi', 0.7647860050201416),
 ('sps', 0.7600197792053223),
 ('fjfi', 0.7560790777206421),
 ('utb', 0.7461121082305908)]
```

In the second case, Word2vec also correctly identified that between cities of the Czech Republic Bratislava is the odd one out.

```
model.wv.doesnt_match(
    "praha brno ostrava bratislava plzen".split())
```

```
'bratislava'
```

In the last case, we tried the vector arithmetic of subtracting a country from its capital and adding another country to get its capital. That also worked reasonably well for Czechia, Prague and Poland.

```
model.most_similar(positive=['praha','polsko'],negative=['cesko'])
```

```
[('varsava', 0.4813214838504791),
 ('fesenko', 0.4719114899635315),
 ('warszawa', 0.47111159563064575),
 ('dablice', 0.46589940786361694),
 ('sztuki', 0.46552157402038574),
 ('budapest', 0.4638131260871887),
 ('kobylisy', 0.462738573551178),
 ('nadr', 0.46272414922714233),
 ('historyczne', 0.45692843198776245),
 ('dziejow', 0.4545629620552063)]
```

3.5.2 Results

In the binary scenario, the Random forest reached the accuracy of **95 %** with the MCC of **0.61** outperforming both Logistic regression and linear SVM in all the metrics as can be seen in table 3.5.

Model	Accuracy	Precision	Recall	F1	MCC
Random Forest	**95 %**	**94 %**	**95 %**	**94 %**	**0.61**
Log Reg	91 %	89 %	91 %	89 %	0.26
LSVM	91 %	87 %	91 %	87 %	0.11

Table 3.5: Accuracy of the Word2vec based classifiers on a binary problem. Bold numbers denote best results in each metric.

In case of the multiclass classification, the Random forest was still on top with solid accuracy of **82 %** and MCC of **0.55** close to the one in the binary case as can be seen in table 3.6. The other models did not perform as well in any of the metrics.

Model	Accuracy	Precision	Recall	F1	MCC
Random Forest	**82 %**	**81 %**	**82 %**	**80 %**	**0.55**
Log Reg	72 %	65 %	72 %	64 %	0.18
LSVM	72 %	64 %	62 %	62 %	0.14

Table 3.6: Accuracy of the Word2vec based classifiers on a multiclass problem. Bold numbers denote best results in each metric.

The confusion matrices 3.5 and 3.6 of the Logistic regression and SVM once again show that both of these models struggled with identifying reviews of a neutral sentiment.

3.6 The BERT scenario

BERT uses a pre-trained model to extract features from input documents in multiple languages. We chose the latest multilingual cased model contains 104 languages including Czech. We used the included Jupyter notebook from [19] as a template for my experiment.

All we had to do was to fine-tune BERT's output layer to perform sentiment analysis. We configured BERT for pooled output used for sentence-level predictions. We then added a new layer with log softmax activation function to serve as a classifier.

Reality \ Prediction	Neg	Neu	Pos
Neg	1135	436	984
Neu	117	4161	3739
Pos	122	1190	24460

Figure 3.4: Confusion matrix for the Random forest multiclass classifier using the Word2vec representation.

Reality \ Prediction	Neg	Neu	Pos
Neg	221	379	1955
Neu	168	903	6946
Pos	148	665	24959

Figure 3.5: Confusion matrix for the Logistic Regression multiclass classifier using the Word2vec representation.

Reality \ Prediction	Neg	Neu	Pos
Neg	13	310	2232
Neu	14	679	7324
Pos	20	427	25325

Figure 3.6: Confusion matrix for the Linear SVM multiclass classifier using the Word2vec representation.

3.6.1 Results

BERT's binary results reached into 95 % in almost all the metrics with the MCC of 0.65. In the case of the mutliclass classification, BERT's accuracy decreased to 81 % having MCC of 0.53. Both can be seen in table 3.7.

Classification	Accuracy	Precision	Recall	F1	MCC
Binary	95 %	94 %	95 %	94 %	0.65
Multiclass	81 %	79 %	80 %	79 %	0.53

Table 3.7: Scores of the BERT based classifiers.

As we can see in the confusion matrix 3.7 BERT also had trouble classifying neutral reviews as the other models did.

3.7 Discussion

In this section we are going to compare the models on account of three metrics: accuracy, F1 score and Matthews correlation coefficient. The accuracy comparison can be seen in table 3.8, the F1 comparison in table 3.9 and the MCC comparison in table 3.10.

The highest accuracy in binary classification was reached by the linear SVM model performed on data represented as TF-IDF and the BERT senti-

Reality \ Prediction	Neg	Neu	Pos
Neg	1463	750	342
Neu	397	3621	3999
Pos	155	1445	24172

Figure 3.7: Confusion matrix for the BERT multiclass classifier.

ment classifier. Both of these models achieved a **96 %** accuracy score. The Random forest in the Word2vec scenario performed only slightly worse with accuracy of 95 % rendering all three representations equally suitable for binary sentiment analysis tasks.

Scenario	**Random forest**	**Log Reg**	**LSVM**
TF-IDF binary	95 %	94 %	**96 %**
TF-IDF multiclass	**84 %**	79 %	81 %
Word2vec binary	95 %	91 %	91 %
Word2vec multiclass	82 %	72 %	72 %
BERT binary	**96 %**		
BERT multiclass	81 %		

Table 3.8: Comparison of accuracy across all experiments. Bold numbers denote best results in binary and multiclass classification respectively.

The differences in multiclass classification are much more noticeable. The overall best model was Random forest with TF-IDF representation followed by the Word2vec version as can be seen in table 3.8. BERT was also not that behind with 81 % accuracy score. On the other hand Logistic regression and LSVM did not perform as well in the Word2vec scenario reaching only 72 % in both cases.

The F1 score mostly mirrors the accuracy in the binary case. The TF-IDF based linear SVM is better than BERT in that department because of its

Scenario	Random forest	Log Reg	LSVM
TF-IDF binary	95 %	93 %	**96 %**
TF-IDF multiclass	**82 %**	77 %	80 %
Word2vec binary	94 %	89 %	87 %
Word2vec multiclass	80 %	64 %	62 %
BERT binary	94 %		
BERT multiclass	79 %		

Table 3.9: Comparison of F1 score across all experiments. Bold numbers denote best results in binary and multiclass classification respectively.

higher precision. In the multiclass, case the highest F1 scores were achieved by the Random forest with both TF-IDF and Word2vec. We can also see the F1 scores of the other Word2vec based models decrease due to their poor precision and recall as can be seen in in table 3.9.

Scenario	Random forest	Log Reg	LSVM
TF-IDF binary	0.67	0.58	**0.73**
TF-IDF multiclass	**0.60**	0.48	0.54
Word2vec binary	0.61	0.26	0.11
Word2vec multiclass	0.55	0.18	0.14
BERT binary	0.65		
BERT multiclass	0.53		

Table 3.10: Comparison of Matthew correlation coeficient across all experiments. Bold numbers denote best results in binary and multiclass classification respectively.

The final metric is Matthews correlation coefficient. It takes into account the imbalance of class distributions in our dataset therefore giving us the truly most accurate model. The highest MCC of **0.73** in the binary case was achieved by linear SVM performed on TF-IDF making it the best method for binary sentiment analysis. In the multiclass scenario the best model was also TF-IDF based Random forest with MCC of **0.60**.

All in all, the TF-IDF representation performed better than the newer methods in all the metrics. It achieved even greater accuracy in the multiclass case than Habernal et. al did in [11]. They achieved 75 % on the same dataset using MaxEnt (Logistic regression) and SVM classifiers while utilizing advanced preprocessing methods based on linguistics while our TF-IDF experiment achieved an accuracy of 84 % using a Random forest and 81 % using the linear SVM. The Word2vec based Random forest and BERT also both performed better with 82 % and 81 % of accuracy respectively.

Conclusion

We have reviewed the state of the art methods of text representations for sentiment analysis. We selected three of those methods and performed experiments with them using the data of Czech product reviews from Mall.cz as an input. Those models were evaluated using a variety of metrics to determine whether they are useful for sentiment analysis in the Czech language.

The traditional TF-IDF based models performed the best out of all three representations. The Word2vec based Random forest had performed similarly well to the TF-IDF models. On the other hand the other Word2vec based models did not achieve any interesting results. BERT achieved similar scores to the other models in binary classification. Despite the BERT classifier being a state of the art method, it did not achieve the performance in the multiple class sentiment analysis as we would expect.

To conclude, all of these outcomes suggest that using state of the art methods for sentiment analysis in the Czech language is a viable strategy. Further research should be conducted in the field of preprocessing because our simple TF-IDF representation achieved higher accuracy than it did in the reference work of [11] without using advanced methods of preprocessing such as lemmatisation. Nonetheless, there is definitely a great potential in BERT as it can be fine-tuned in countless ways to better suit the sentiment analysis needs in the Czech environment.

Bibliography

1. HATZIVASSILOGLOU, Vasileios; MCKEOWN, Kathleen R. Predicting the Semantic Orientation of Adjectives. In: *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*. Madrid, Spain: Association for Computational Linguistics, 1997, pp. 174–181. ACL '98/EACL '98. Available from DOI: 10.3115/976909.979640.
2. TURNEY, Peter D; LITTMAN, Michael L. Unsupervised learning of semantic orientation from a hundred-billion-word corpus. *arXiv preprint cs/0212012*. 2002.
3. PANG, Bo; LEE, Lillian; VAITHYANATHAN, Shivakumar. Thumbs Up?: Sentiment Classification Using Machine Learning Techniques. In: *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 79–86. EMNLP '02. Available from DOI: 10.3115/1118693.1118704.
4. PANG, Bo; LEE, Lillian. *Movie Review Data* [online]. 2004 [visited on 2019-04-15]. Available from: http://www.cs.cornell.edu/people/pabo/movie-review-data/ Original 2002 dataset is not accessible anymore.
5. MIKOLOV, Tomas; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. Efficient Estimation of Word Representations in Vector Space. In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. 2013. Available also from: http://arxiv.org/abs/1301.3781.
6. ZHANG, Dongwen; XU, Hua; SU, Zengcai; XU, Yunfeng. Chinese comments sentiment classification based on word2vec and SVMperf. *Expert Systems with Applications*. 2015, vol. 42, no. 4, pp. 1857–1863. ISSN

0957-4174. Available from DOI: https://doi.org/10.1016/j.eswa.2014.09.011.

7. DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805*. 2018.
8. MEDHAT, Walaa; HASSAN, Ahmed; KORASHY, Hoda. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*. 2014, vol. 5, no. 4, pp. 1093–1113. ISSN 2090-4479. Available from DOI: https://doi.org/10.1016/j.asej.2014.04.011.
9. VESELOVSKÁ, Katerina; HAJIC, Jan; SINDLEROVÁ, Jana. Creating annotated resources for polarity classification in Czech. In: *KONVENS*. 2012, pp. 296–304.
10. STEINBERGER, Josef et al. Creating sentiment dictionaries via triangulation. In: *Proceedings of the 2nd workshop on computational approaches to subjectivity and sentiment analysis*. 2011, pp. 28–36.
11. HABERNAL, Ivan; PTÁČEK, Tomáš; STEINBERGER, Josef. Sentiment Analysis in Czech Social Media Using Supervised Machine Learning. In: *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*. Atlanta, Georgia: Association for Computational Linguistics, 2013, pp. 65–74. Available also from: http://www.aclweb.org/anthology/W13-1609.
12. BREIMAN, Leo. Random Forests. *Machine Learning*. 2001, vol. 45, no. 1, pp. 5–32. ISSN 1573-0565. Available from DOI: 10.1023/A:1010933404324.
13. QUINLAN, J. Ross. *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993. ISBN 1-55860-238-0.
14. WWW.MACHINEINTELLEGENCE.COM. *Iris Entropy Max Depth* [online] [visited on 2019-04-25]. Available from: http://www.machineintellegence.com/wp-content/uploads/2018/04/iris_entropy_MaxDepth.png [online image].
15. BISHOP, Christopher M. *Pattern recognition and machine learning, 5th Edition*. Springer, 2007. Information science and statistics. ISBN 9780387310732. Available also from: http://www.worldcat.org/oclc/71008143.
16. PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, vol. 12, pp. 2825–2830.
17. KAREL KLOUDA Juan Pablo Maldonado Lopez, Daniel Vašata. *BI-VZD Přednáška 12: Zpracování přirozeného jazyka* [online]. 2018 [visited on 2019-04-19]. Available from: https://courses.fit.cvut.cz/BI-VZD/lectures/files/BI-VZD-12-cs-slides.pdf. The file is available after logging into the ČVUT network - copy of the file is saved in the enclosed USB drive.

18. *Vector Representations of Words* [online] [visited on 2019-05-12]. Available from: https://www.tensorflow.org/tutorials/representation/word2vec.

19. DEVLIN, Jacob. *TensorFlow code and pre-trained models for BERT* [online] [visited on 2019-04-18]. Available from: https://github.com/google-research/bert. [online].

20. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N; KAISER, Łukasz; POLOSUKHIN, Illia. Attention is all you need. In: *Advances in neural information processing systems*. 2017, pp. 5998–6008.

21. ALLARD, Maxime. *What is a Transformer?* [online] [visited on 2019-05-06]. Available from: https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04.

22. CALVO, Miguel Romero. *Dissecting BERT Part 1: The Encoder* [online] [visited on 2019-05-10]. Available from: https://medium.com/dissecting-bert/dissecting-bert-part-1-d3c3d495cdb3.

23. BEHNEL, S.; BRADSHAW, R.; CITRO, C.; DALCIN, L.; SELJEBOTN, D.S.; SMITH, K. Cython: The Best of Both Worlds. *Computing in Science Engineering*. 2011, vol. 13, no. 2, pp. 31–39. ISSN 1521-9615. Available from DOI: 10.1109/MCSE.2010.118.

24. ŘEHŮŘEK, Radim; SOJKA, Petr. Software Framework for Topic Modelling with Large Corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, 2010, pp. 45–50. http://is.muni.cz/publication/884893/en.

25. TECHNOLOGIES, RaRe. *Topic Modelling for Humans* [online] [visited on 2019-04-06]. Available from: https://github.com/RaRe-Technologies/gensim#adopters.

26. PÉREZ, Fernando; GRANGER, Brian E. IPython: a System for Interactive Scientific Computing. *Computing in Science and Engineering*. 2007, vol. 9, no. 3, pp. 21–29. ISSN 1521-9615. Available from DOI: 10.1109/MCSE.2007.53.

27. *Kronika české Wikipedie* [online] [visited on 2019-04-18]. Available from: https://cs.wikipedia.org/wiki/Wikipedie:Kronika [online].

28. BURKE, Sean M. *Unidecode* [online] [visited on 2019-04-18]. Available from: https://pypi.org/project/Unidecode/#description. [software].

29. KOPPEL, Moshe; SCHLER, Jonathan. The importance of neutral examples for learning sentiment. *Computational Intelligence*. 2006, vol. 22, no. 2, pp. 100–109.

30. VARUN, Divya. *Sentiment analysis using word2vec* [online] [visited on 2019-05-07]. Available from: `https://www.kaggle.com/varun08/sentiment-analysis-using-word2vec`.

Appendix A

Examples

A.1 TF-IDF multiclass classification

Table A.1: 15 sample reviews with predictions and real sentiment values. Classified by Random forest with TF-IDF representation.

Review	Prediction	Reality
Granule velmi dobré kvality. Srst uzasna, pejskovi taky chutnají.	Positive	Positive
Continued on next page		

Table A.1 – continued from previous page

Review	Prediction	Reality
Myslela jsem si, že je to klasická velká myš, ale pravda je opakem. Mě konkrétně to nevadí. Myš má navíjecí kolečko na kabel, takže Vám nikde nepřekáží. Myš samotná je super, nemůžu si na ni stěžovat. Je příjemná na omak, je i designově hezká. Je velmi dobrá k práci. Všem doporučuji	Positive	Positive
Slaby bass, jinak vyborna sluchatka.	Negative	Neutral
bez problému a za výbornou cenu:-)	Positive	Positive
S mobilním telefonem Samsung S5230 Star Pink jsem velmi spokojená a doporučuji ho všem. V mém okolí už se tak stalo a koupili si ho tři známí. Dělá krásné fotky, má mnoho zajímavých funkcí a je jednoduchý na ovládání.	Positive	Positive
Nejprve přišel tester s vybitou baterkou a evidentně použitým hrudním pásem, takže se dostavilo zklamání. Nicméně reklamace proběhla bez potíží. Nyní přístroj funguje , jak má. Spokojenost.	Neutral	Neutral
Spokojenost. Odolný, spolehlivý, hezký a bezporuchový USB flash disk.	Positive	Positive
Velmi variabilní a stabilní, jednoduché, kvalitní zpracování, vše pro montáž v balení	Positive	Positive
Zatím jsem upekla jen několik chlebů, nezkoušela jsem výrobu džemu či těsta, pekárnu si ale nemůžu vynachválit. Šetří čas, snadno se myje, není příliš hlučná. Výborný poměr cena/výkon.	Positive	Positive

Continued on next page

Table A.1 – continued from previous page

Review	Prediction	Reality
Toastovač robusnější, ale lehký a pěkný a funguje skvěle.	Positive	Positive
Barva hezká, velikost odpovídá. Jen trochu pouští chlupy, snad se to poddá.	Positive	Neutral
Jsem moc spokojená.Jak s kvalitou plenek,tak s cenou a rychlostí dodání na mall.cz.	Positive	Positive
Spokojenost, svěží mladistvá vůně, mohla by déle vydržet	Neutral	Neutral
Strašně sladké!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!	Negative	Negative
Chutnají mu a to je hlavní	Positive	Positive

A.2 Word2vec multiclass classification

Table A.2: 15 sample reviews with predictions and real sentiment values. Classified by Random forest with Word2vec based representation.

Review	Prediction	Reality
Granule velmi dobré kvality. Srst uzasna, pejskovi taky chutnají.	Positive	Positive
Myslela jsem si, že je to klasická velká myš, ale pravda je opakem. Mě konkrétně to nevadí. Myš má navíjecí kolečko na kabel, takže Vám nikde nepřekáží. Myš samotná je super, nemůžu si na ni stěžovat. Je příjemná na omak, je i designově hezká. Je velmi dobrá k práci. Všem doporučuji	Positive	Positive
Slaby bass, jinak vyborna sluchatka.	Positive	Neutral
bez problému a za výbornou cenu:-)	Positive	Positive
S mobilním telefonem Samsung S5230 Star Pink jsem velmi spokojená a doporučuji ho všem. V mém okolí už se tak stalo a koupili si ho tři známí. Dělá krásné fotky, má mnoho zajímavých funkcí a je jednoduchý na ovládání.	Positive	Positive
Nejprve přišel tester s vybitou baterkou a evidentně použitým hrudním pásem, takže se dostavilo zklamání. Nicméně reklamace proběhla bez potíží. Nyní přístroj funguje , jak má. Spokojenost.	Positive	Neutral
Spokojenost. Odolný, spolehlivý, hezký a bezporuchový USB flash disk.	Positive	Positive

Continued on next page

Table A.2 – continued from previous page

Review	Prediction	Reality
Velmi variabilní a stabilní, jednoduché, kvalitní zpracování, vše pro montáž v balení	Positive	Positive
Zatím jsem upěkla jen několik chlebů, nezkoušela jsem výrobu džemu či těsta, pekárnu si ale nemůžu vynachválit. Šetří čas, snadno se myje, není příliš hlučná. Výborný poměr cena/výkon.	Neutral	Positive
Toastovač robusnější, ale lehký a pěkný a funguje skvěle.	Positive	Positive
Barva hezká, velikost odpovídá. Jen trochu pouští chlupy, snad se to poddá.	Neutral	Neutral
Jsem moc spokojená.Jak s kvalitou plenek,tak s cenou a rychlostí dodání na mall.cz.	Positive	Positive
Spokojenost, svěží mladistvá vůně, mohla by déle vydržet	Neutral	Neutral
Strašně sladké!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!	Negative	Negative
Chutnají mu a to je hlavní	Positive	Positive

A.3 BERT multiclass classification

Table A.3: 15 sample reviews with predictions and real sentiment values. Classified by BERT.

Review	Prediction	Reality
"Granule velmi dobré kvality. Srst uzasna, pejskovi taky chutnají."	Positive	Positive
"Myslela jsem si, že je to klasická velká myš, ale pravda je opakem. Mě konkrétně to nevadí. Myš má navíjecí kolečko na kabel, takže Vám nikde nepřekáží. Myš samotná je super, nemůžu si na ni stěžovat. Je příjemná na omak, je i designově hezká. Je velmi dobrá k práci. Všem doporučuji"	Positive	Positive
"Slaby bass, jinak vyborna sluchatka."	Positive	Neutral
"bez problému a za výbornou cenu:-)"	Positive	Positive
"S mobilním telefonem Samsung S5230 Star Pink jsem velmi spokojená a doporučuji ho všem. V mém okolí už se tak stalo a koupili si ho tři známí. Dělá krásné fotky, má mnoho zajímavých funkcí a je jednoduchý na ovládání."	Positive	Positive
"Nejprve přišel tester s vybitou baterkou a evidentně použitým hrudním pásem, takže se dostavilo zklamání. Nicméně reklamace proběhla bez potíží. Nyní přístroj funguje , jak má. Spokojenost."	Neutral	Neutral
"Spokojenost. Odolný, spolehlivý, hezký a bezporuchový USB flash disk."	Positive	Positive

Continued on next page

Table A.3 – continued from previous page

Review	Prediction	Reality
”Velmi variabilní a stabilní, jednoduché, kvalitní zpracování, vše pro montáž v balení”	Positive	Positive
”Zatím jsem upěkla jen několik chlebů, nezkoušela jsem výrobu džemu či těsta, pekárnu si ale nemůžu vynachválit. Šetří čas, snadno se myje, není příliš hlučná. Výborný poměr cena/výkon.”	Positive	Positive
”Toastovač robusnější, ale lehký a pěkný a funguje skvěle.”	Positive	Positive
”Barva hezká, velikost odpovídá. Jen trochu pouští chlupy, snad se to poddá.”	Neutral	Neutral
”Jsem moc spokojená.Jak s kvalitou plenek,tak s cenou a rychlostí dodání na mall.cz.”	Positive	Positive
”Spokojenost, svěží mladistvá vůně, mohla by déle vydržet”	Neutral	Neutral
”Strašně sladké!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!”	Negative	Negative
”Chutnají mu a to je hlavní”	Positive	Positive

Appendix B

Acronyms

NLP Natural Language Processing

ML Machine Learning

MaxEnt Maximum Entropy

SVM Support Vector Machines

MAP Maximum A Posteriori

NN Neural Network

TF-IDF Term Frequency – Inverse Document Frequency

TP True Positives

FP False Positives

TN True Negatives

FN False Negatives

MCC Matthews Correlation Coefficient

Appendix C

Contents of enclosed CD

- `readme.md` ..... the file with CD contents description
- `bert` ..... the directory with the notebook for the BERT experiment
- `data` ..... the directory of input data
  - `mallcz` ..... the directory containing the Mall.cz dataset
- `doc` ..... the directory of LaTeX source codes of the thesis
  - `assets` ..... the directory of assets used in the thesis
- `tf-idf` ..... the directory with the notebook for the TF-IDF experiment
- `word2vec` .. the directory with the notebook for the Word2vec experiment