TECHNISCHE UNIVERSITÄT DRESDEN

Master Thesis

Sentiment analysis of contexts around query terms in documents

Kateryna Rybina

Course of studying: Computational Engineering

Institute for Systems Architecture
Chair of Computer Networks

Professor: Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill
Supervisor: Dipl.-Medien-Inf. David Urbansky
Working time: 15.07.2011 to 14.01.2012

Declaration of Authorship

In the following I, Kateryna Rybina, want to state that this thesis titled “Sentiment analysis of contexts around query terms in documents” and the content presented in it are results of my own work.

All published works of others, which were utilized in this thesis, are referenced and can be found in the Bibliography. All direct or indirect thoughts taken from published external resources were marked and put to the Bibliography. The Palladian toolkit upon which the classification algorithms were developed is considered in the Bibliography as well.

Dresden 14.01.2012

Contents

List of Figures

List of Tables

Chapter 1: Introduction

For several years we can observe a rapid growth of World Wide Web users. According to the Internet usage statistics[1] the amount of Internet users worldwide increased by 480.4% during 2000-2011. Users share their experiences, publish opinions, communicate, consume, and spread a huge amount of user-generated content. Especially opinionated content is of a great value for companies whose products and services are being discussed and assessed. As a reaction, various mechanisms and systems have been developed to derive opinionated content from the web and to analyze it.

An extensive amount of research directed to documents' opinion classification techniques has been done for English language but approaches and algorithms for German language sentiment classification are barely present in academic literature. It thus remains a huge field for research efforts. Even though numerous companies dealing with the provision of opinion mining services for several languages - including German language - do exist, they keep the implementation details of the utilized sentiment analysis algorithms and the quality of their approaches closed. This thesis intends to address this issue by providing an in-depth analysis of two German sentiment classification algorithms. In the following a short motivation for the associated research field is provided. Furthermore, the central research goals addressed by this thesis are derived and an overview of the structure of the following chapters is given.

1.1 Motivation of the research field

Nowadays, people are extensively utilizing various social media communication means: they create online accounts in social networks to keep in touch with friends and business partners, run and read blogs in order to share opinions, obtain important information, advertise and sell their products online.

The simplicity and low-cost of the usage of communication means allows customers to publish their experiences and opinions about the purchased products and services online by creating a blog post or giving a positive or negative assessment in products' review portals. The increased amount of opinionated content on the Internet is highly important for big companies as a good company image is an important factor in influencing customer's purchasing behavior and preferences (David Joshua Perdue, 2010). The Universal McCann's research, dedicated to analyzing the consumer usage of social media platforms shows that 72,8% of interviewed active Internet users read blogs which by its nature contain a lot of opinionated content, 36% of interviewed people think more positively about companies that have blogs and 34% post opinions about products and brands on their blogs (Tom Smith, 2008). These figures show the importance of this field for the industrial world.

Big companies pay so much attention to their reputation that they hire other companies in order to keep track of the company image (David Joshua Perdue, 2010). That is why highly efficient media monitoring services capable of performing sentiment analysis are of a great importance to the companies dealing with analyzing and monitoring opinionated content in the World Wide Web.

[1]Internet World Stats: http://www.internetworldstats.com/stats.htm.

The research field into which this work is embedded – sentiment analysis in the Web - is motivated by two types of application areas which are affected by opinionated content in the Web: Business-to-Business (B2B) and Consumer-to-Business (C2B). These two areas are shortly discussed by generic use cases in the following.

1.1.1 Business-to-Business use-case

The B2B use case emphasizes the importance for companies to care for how they are perceived in the Web. Let us assume that the company **A** wants to invest a big sum of money in one of the competing companies **B** or **C**. It is important for company **A** to make a correct decision as the successful investment will double its budget and a failure can lead to weakening its position on the market. This is the rationale for managers of the enterprise **A** to collect and carefully analyze information of both competing companies. Enterprises **B** and **C** produce the same type of products which could satisfy the needs of the investing company. At the presence of equal conditions the choice of an investor will be done in favor of the company with a better reputation, the company whose products are assessed positively by users in the web, the company which is more reactive and responsive to customers' needs and problems.

1.1.2 Consumer-to-Business use-case

The C2B use case shows that the sentiment analysis system could also be beneficial for the Internet users that want to get the information about the product they are willing to purchase. People should not spend a lot of time by visiting different web sites: blogs, review portals in order to obtain specific information they are searching for. They can minimize the effort and save time using the sentiment analysis system which will return the opinionated content from different web sources on their behalf.

The two mentioned use cases show the importance of sentiment analysis in the Web. Due to this, various systems have been developed which are capable of classifying sentences into categories of sentiments. Unfortunately, currently no detailed analysis is available about the classification performance of German sentiment classifiers. This gap is intended to be closed by this thesis.

1.2 Goals and Research questions

The aim of this master thesis is to compare the performance of two sentiment classifiers of German language content, namely the Palladian Text Classifier (PTC) and the Palladian Sentiment Classifier (PSC) and to analyze which of them shows better classification performance. The former uses supervised learning technique for document's sentiment classification. The latter utilizes a lexicon-based approach and uses for the classification task SentimentWortschatz[2], a dictionary of German language sentiment words. Different features are going to be applied to these classifiers in order to obtain the best classifiers' configurations which give the best evaluation results. The classifiers' performance will be tested on three datasets derived from different domains, namely Amazon product reviews and a variety of Internet resources (web blogs, news portals, chatting portals, Wikipedia). The classification will be done on the sentence level and classifiers are going to distinguish which type of opinion - positive, negative, or neutral – express the input sentences. It will be analyzed if the performance results of these classifiers are skewed towards any sentence tonality and whether they are domain independent. The prototype for two classifiers that implement tonality

[2]URL: http://wortschatz.informatik.uni-leipzig.de/download/sentiws.html

analysis will be developed, before, as a last step, their performances will be evaluated as well as compared.

Besides the general analysis, the thesis intends to determine the best configuration of features for Palladian Text Classifier and Palladian Sentiment Classifier. Thus, researchers can reuse the achieved results and embed the best configurations of PTC or PSC into an opinion mining system for German language documents' analysis. In order to realize this, a set of valid features and classification algorithms will be established for both researched classifiers. Figure 1.1 shows the summary of the master thesis goal, namely the analysis and evaluation of the PTC and PSC classifiers which can be embedded into any opinion mining system for German language. The input for these classifiers are the sentences obtained from the different resources in WWW while in the output these sentences must be classified as carrying positive, neutral, or negative opinions.

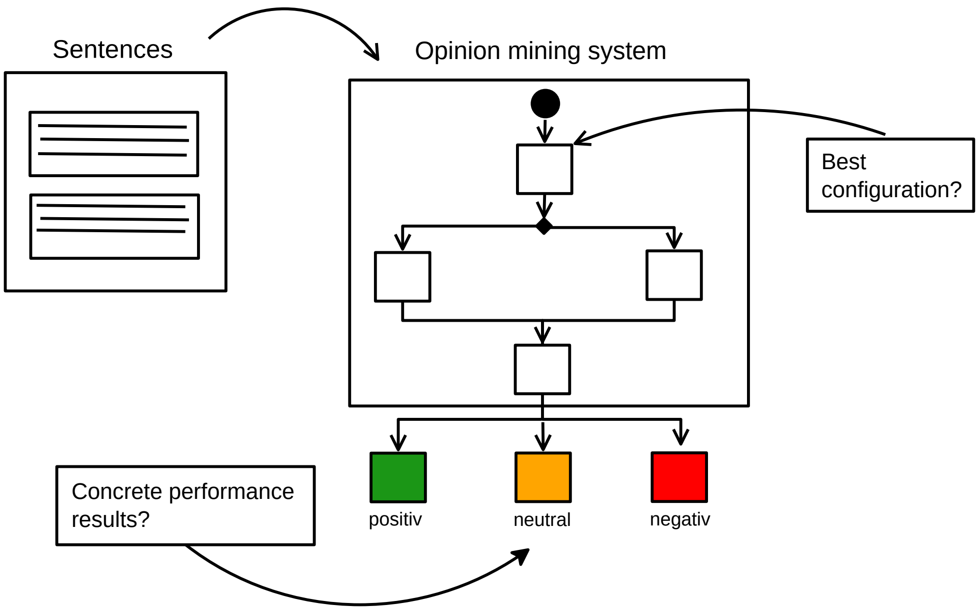


Figure 1.1: Summary of the master thesis goal.

More precise, this master thesis will explore and address the following research questions:

1. Is the lexicon-based Palladian Sentiment Classifier better than Palladian Text Classifier based on supervised-learning?
2. Which features should be applied to each classifier in order to achieve the best classification performance?
3. Are the classifiers' performances skewed towards a particular sentence tonality?
4. Is PTC's and PSC's classification behavior domain specific?

By answering these questions, a detailed analysis of the quality and characteristics of the two focused classifiers is provided.

1.3. Structure

The overall thesis is structured as following:

Chapter 2 presents central background information as well as a detailed state-of-the-art analysis of existing commercial and academic sentiment analysis systems and the opinion classification algorithms applied. The strengths and weaknesses of existing systems are reviewed and emphasized. This analysis emphasizes the gap that will be closed by this master thesis. Chapter 3 introduces both, the concepts of PTC and PSC and their implementation. It introduces the experimental setup for each classifier, describes the developed classification algorithms, and provides the implementation details of both PTC and PSC. Chapter 4 presents evaluation results of the applied research. After three datasets have been introduced in Chapter 4, the optimal dataset characteristic for learning PTC is identified. Then, the classifiers performances are evaluated and all research questions are discussed individually. Chapter 5 concludes the master thesis and provides an outlook on the future work.

Chapter 2: Background and state-of-the-art

In this chapter background information which is fundamental for understanding the work accomplished in this master thesis is presented. First of all an introduction to the target research area is given in Section 2.1. Afterwards, the features which might be used for sentiment analysis are presented in Section 2.2. Section 2.3 describes the metrics of performance evaluation used in sentiment analysis, and thus in this master thesis.
Due to the fact that one of the classifiers implemented in this master thesis, namely Palladian Sentiment Classifier is lexicon-based, the principles of lexicon generation are presented in Section 2.4 and publicly available German-language resources for sentiment analysis are described in Section 2.5.
The second classifier, namely Palladian Text Classifier uses a classification approach based on supervised learning. That is why Section 2.6 gives a brief introduction to machine learning and Section 2.7 describes supervised learning approaches. In Section 2.8 the core principles of unsupervised learning are presented. Section 2.9 presents state-of-the art of sentiment analysis systems with the main emphasize on the sentiment classification algorithms applied. Section 10 summarizes information and concludes this chapter.

2.1 Opinion mining

The idea of automatic extraction, classification, and definition of opinion polarity of documents or sentences can be verbalized as *opinion mining, sentiment analysis,* or *subjectivity analysis*. These terms will be used synonymously in the following chapters. In general, text documents can be split into two broad categories which are objective and subjective. It means that the *objective document* (sentence) expresses some factual information on a given object while *subjective document* (sentence) expresses personal feeling or beliefs in regards to this object. The *opinionated sentence* is such a sentence that expresses explicit or implicit positive or negative opinion e.g. sentences in a product review. The o*pinion polarity, sentiment tonality*, or *orientation of the opinion* means in most cases whether a document (sentence) expresses positive, negative, or neutral sentiment (Bing Liu, 2010).

When talking about opinion mining, it is important to understand what makes the sentence opinionated, which techniques can be used to automatically extract and classify sentences as being opinionated or objective, and in some applications as well to determine the opinion holder.

2.2 Sentiment analysis features

In this thesis the term feature in relation to sentiment analysis is defined as an indicator of opinion in the document (sentence). So having a sufficient set of features in the document can be a good indicator that it is a subjective document. These selected features are used in a next step as an input for the classifier.

It is an often used technique in information retrieval (IR) to represent a piece of text as a feature vector wherein the entries correspond to individual terms. There are numerous features that can be applied for sentiment analysis with the aim to classify documents (sentences) as being opinionated. Some of these features were analysed in the sentiment analysis overview (Pang, 2008) and are briefly summarized and described below.

2.2.1 Term presence vs. frequency

It was discovered by (Pang, 2008) that for the opinion mining classification task the presence of terms is more important than their frequency, which is the opposite way for the topic relevance classification. It means that a binary-valued feature vectors, where the entries indicate if the term occurs (value 1) or not (value 0) are more efficient for review polarity classification than real-valued feature vectors in which entry value increase when the frequency of the occurrence of the term increase.

2.2.2 Position information

The position of a token in a text unit (e.g. in the beginning, middle or at the end) may play an important role on how this token affect the overall sentiment of this text unit. That is why the position information is sometimes encoded into the feature vectors.

2.2.3 Syntax

Including syntactic relations into the feature sets was done by some of the researchers. This deep linguistic analysis can be beneficial for text classification. As an example the use of higher-order n-grams and dependency can be considered for document-level classification. The usage of syntactic patterns can be beneficial for the classification task. For example, in the sentence “This music is nice and charming. ” and the German variant “Diese Musik ist schön und charmant.” the conjunction and (und) define that if given that the first adjective is a positive sentiment word, the second adjective must also be positive. These rules are also defined for the other connectives such as or, but, either or etc. Parsing the text can serve as basis for modelling negations and intensifiers (Kennedy, 2006). Collocations and more complex syntactic patterns also appeared to be useful for the detection of subjectivity (Riloff, 2003; Wiebe, 2004).

2.2.4 N-grams

N-grams are sets of tokens of the length n. In some applications (e.g. defining product review’s polarity) bigrams and trigrams show better polarity classification. Two main types of n-grams are: *character level n-grams* and *word level n-grams.* Character level n-grams use each character of the string as a token. Word level n-grams use each word of the string as a token. The number of n-grams in a set can be calculated as following (Urbansky, 2010):

$$Ngrams = NumberOfTokens - n + 1 \qquad (1)$$

2.2.5 Contrastive distance between terms

“Contrastive distance” between terms can be considered as a further feature too. Snyder (2007) used a pair of contrastive with their opinion polarity words “delicious” and “dirty” as an automatically computed feature and as a part of a rating-inference system (Pang, 2008).

2.2.6 Parts of speech

Parts of speech (POS) tagging is often used in opinion mining applications. It was discovered that for English language adjectives, adverbs, verbs, and nouns are parts of speech that are carrying sentiment (Benamara, 2007). It was proved that adjectives are the best indicators of subjectivity which is the reason why they are often used as core features for classification task (Pang, 2008).

2.2.7 Negation

It is very important to detect negations in opinion mining applications as only one token may change the polarity of the sentence and thus result into classifying this sentence into another category. For example in the sentences “Das Piano klingt gut.” and “Das Piano klingt nicht gut.” the only token “nicht” is different and it will result in classifying the sentence into the negative category. Some researches encode negations directly into the definitions of the initial features (Das, 2001), others deal with negations as a second-order feature of a text segment. In the latter approach the feature vector initially ignores negations but then the representation is converted into a different representation that is negation aware (Pang, 2008).

2.3 Evaluating the performance of the classification algorithm

The main evaluation criterion for classifiers is accuracy. The accuracy of a classification model on a test set is defined as (Bing Liu, 2007):

$$Accuracy = \frac{Number\ \ of\ correct\ \ classifications}{Total\ \ number\ \ of\ test\ cases}, \qquad (2)$$

where the correct classification means that the model predicts the same class as the test data was labelled with.

Precision and recall are two other classifier performance measures. The former is a measure of exactness while the latter is a measure of completeness of classification on the positive class (the class that the user is interested in). In opinion retrieval precision takes into consideration all the retrieved documents, but it can be also evaluated when ranking of the documents is taken into consideration and when only the topmost results are considered e.g. k topmost documents. It is then called precision at rank k or *P@k* (Markov, 2007).

Precision and recall can be introduced by a confusion matrix that contains information about actual and predicted results from classifier that is shown on Table 2.1.

	Classified positive	Classified negative
Actual positive	TP	FN
Actual negative	FP	TN

Table 2.1: Confusion matrix of classifier performance (Bing Liu, 2007, Table 3.2).

In this matrix *TP* means the number of correct classification of the positive examples (true positive),
FN is the number of incorrect classifications of positive examples (false negative),
FP is the number negative examples which are incorrectly classified as positive (false positive), and
TN is the number of correct classifications of negative examples (true negative).
Based on the confusion matrix the formulas for precision and recall can be written as the following.

$$p = \frac{TP}{TP+FP}, \qquad r = \frac{TP}{TP+FN} \qquad (3), (4)$$

It is reasonable in practice to combine these two measures as the high precision is usually received by the cost of low recall and vice versa. The metric of classifier performance that combines both precision and recall is called F-measure. In case when precision and recall are evenly weighted we obtain F_1-score that is calculated as a harmonic mean of precision *p* and recall *r*.

$$F_1 = \frac{2\,pr}{p+r} \qquad (5)$$

F_1- score is a general case of F_β –measure, where the user may put more importance on either precision or recall, depending on the value of β (van Rijsbergen, 1979).

$$F_\beta = (1 + \beta^2) * \frac{pr}{\beta^2 p + r} \qquad (6)$$

Cross-validation is the common method for classifier evaluation. It can also be used when the data set is small. The idea of *n-fold cross-validation* is that the data set must be partitioned into n equal-size disjoint subsets. Each subset is used as the training set and the remaining n - 1 subsets are combined as the test set. The result gives n accuracies for n data sets and the final estimated accuracy is the average of the n accuracies. 10-fold and 5-fold cross validation are usually used. Another variant of cross-validation is the *leave-one-out cross-validation*, which is particularly beneficial for very small data sets. All the data examples are used for training the classifier and only one test example for evaluation. This type of cross-validation is not efficient for large data sets as for the *m* training examples *m* classifiers need to be built. (Bing Liu, 2007). For better classification results the ***stratification*** of the data set is preferable. It means that all the data classes should be present in the same proportion in all training and testing data subsets (Markov, 2007).

2.4 Opinion lexicon generation

Before realizing the document (sentence) classification task in many approaches it is important to first establish an opinion lexicon. It means generating opinion lexicon automatically, creating manually or combining those both techniques. With aim to classify documents in the positive, negative, and neutral category, the examples of the positive German sentiment words could be *gut, schön, richtig, exzellent, positiv, glücklich, phantastisch, lieb* etc. The examples of German negative sentiment words could be *schlecht, unschön, falsch, unglücklich, negativ, böse, zweitklassig, armselig, mies* etc. There are three main approaches for collecting sentiment words. They are *manual approaches, dictionary-bases approaches,* and *corpus-based approaches* (Bing Liu, 2010). The manual approach is, however, very time consuming which often leads it to be implemented in combination with one of the below discussed automated approaches.

2.4.1 Dictionary-based approach

The idea of *dictionary-based approach* is first to collect some small set of seed words with their known opinion polarities and then to iteratively extend them with the help of online dictionary e.g. WordNet. The synonyms and antonyms of the opinion seed words are being iteratively searched and added to the seed set and the process stops when no more words could be found. This approach could be combined with the manual inspection of mistakes. The main shortcoming of this approach is that it is not possible to establish with it opinion words with domain specific orientation (Bing Liu, 2010).

2.4.2 Corpus-based approach

This shortcoming can be solved by the *corpus-based approach.* This approach uses as its bases the syntactic or co-occurrence patterns (see Section 2.2.3) and the predefined set of seed words with its polarities. But this approach is more suitable for domain specific lexicon generation as it is very hard to prepare the huge corpus of German (English) words. For identification of all opinion words dictionary-based approach is more suitable (Bing Liu, 2010).

There exist a lot of publicly available online dictionaries for sentiment analysis for English language among which the most famous is WordNet (Fellbaum, 1998). Different possibilities to extend seed sets of words with WordNet are at disposal: some of them are by traversing WordNet relations, others by propagating the seed words to synonym sets (synsets), or by iterating through the WordNet synsets and assigning scores to them with the help of classifiers. SentiWordNet is a widely-used English sentiment lexical resource that was derived this way. SentiWordNet was generated by automatically annotating the synsets of the WordNet, where each synset received the three scores indicating to which extend the respective synset is to be regarded positive, negative or objective. The sum of all three scores always equals one (Esuli and Sebastiani, 2006).

2.4.3 Example of sentiment lexicon generation algorithms

Bas Heerschop (2011) considered several algorithms for creating sentiment lexicon and used WordNet as publicly available resource for English language sentiment analysis. By the means of the first algorithm the semantic lexicon was established by propagating the sentiment of a seed set of words through WordNet semantic relations (WordNetRel). Using another two algorithms (PageRankSeed and PageRankSWN) the sentiment lexicon was created by propagating the sentiment of a seed set to the synsets that are similar to predefined positive and negative synsets. Google PageRank algorithm (Bring, 1998) can be used to exploit the semantic structure of WordNet to create a ranking of how closely synsets relate to positive or negative synsets. And using the fourth (SentiWordNet) algorithm the opinion lexicon is composed by iterating over WordNet synsets and their associated glosses and assigning sentiment scores to these synsets by means of classifier (Bas Heerschop,2011). The authors suggest a simple lexicon-based opinion classifier for investigating the performance of above mentioned algorithms. The classifier is focused on adjectives, adverbs, verbs and nouns. The document score is computed based on the scores of all its sentences and the sentence score is calculated based on the scores of all sentiment-baring words within it. The document is classified to the positive class if this score is equals or is higher than zero; otherwise it is classified as being negative.

The performance of above-mentioned approaches for sentiment lexicon creation was evaluated on the corpus of 1.000 positive and 1.000 negative English movie reviews, which were extracted together with their numerical review scores. For both positive and negative documents the performance was estimated in measure of the precision, recall and F_1-mesure (see formulas in Section 2.3). In general, the performance of the classifier, applying the lexicon generated using all four algorithms appeared to show better results for the classification of positive documents. The authors explain it by the fact that people in writing reviews try to omit the negative words. WordNetRel algorithm for sentiment lexicon creation showed an overall (for classification of both positive and negative documents) F_1-mesure of 41.3% and provided accuracy of 51.9%. PageRankSWN algorithm shows higher recall and precision for classification of the negative documents than PageRankSeed algorithm but it is conversely for the classification of the positive documents. Both PageRank algorithms exhibited the accuracy of 49.7%. SentiWordNet showed slightly biased performance on the corpus provided by authors, since more documents are classified as being positive than negative. This algorithm showed the accuracy of 59.5%. In terms of accuracy SentiWordNet outperforms all other approaches and PageRank-based sentiment propagation method bootstrapped using SentiWordNet score is more robust approach due to the fact that its difference of F_1-mesure between positive and negative documents is smaller than in all other considered approaches (Bas Heerschop,2011).

2.5 German-language resources for sentiment analysis

There has already been done some research in establishing German language online dictionaries of sentiment words and among them two publicly available resources for opinion mining are: *SentimentWortschatz* (*SentiWS*, Remus, 2010) and *GermanPolarityClues* (Waltinger, 2010).

2.5.1 SentiWS

SentiWS lists positive and negative German sentiment bearing words, which are weighted within the interval [-1, 1], their POS and inflections. SentiWS contains 1.650 negative and 1.818 positive sentiment words, which result in 16.406 positive and 16.328 negative word forms. The resource consists of adjectives, adverbs, verbs, and nouns. For assembling SentiWS three sources of information were used: General Inquirer (GI) lexicon, co-occurrence analysis of rated product reviews, and German Collocation Dictionary (Remus, 2010).
GI's positive and negative categories were translated with Google Translator and afterwards manually revised. Co-occurrence analysis was used with the purpose of selecting a list of words which often co-occur in reviews within positively or negatively marked document (positive and negative marker). German Collocation Dictionary contains 25.288 semantic groups with about 6.932 groups related to sentiment and 76 groups strongly related to sentiment from which some new sentiment bearing words were obtained (Remus, 2010).
The weights of the words were received by applying Pointwise Mutual Information (PMI) approach (Church and Hanks, 1990). The idea is to obtain semantic orientation (SO) of the word by its semantic association *A.* Where *SO* of a word *w* is calculated as its association with a manually selected positive set of seed words *P* and manually selected negative set of seed words *N* as shown in Equation 7 .

$$SO(w) = \sum_{p \in P} A(w,p) - \sum_{n \in N} A(w,n) \quad (7)$$

The words *w* has a positive semantic orientation if its *SO(w)* is positive. The semantic association *A(w,p)* and *A(w,n)* are calculated using *PMI* as shown in Equation 8.

$$PMI\ (w_1, w_2) = log_2(\frac{P(w_1 \& w_2)}{P(w_1) * P(w_2)}) \quad (8)$$

Where *P(w)* is the probability that the word *w* occurs and $P(w_1 \& w_2)$ is the probability that w_1 and w_2 co-occure.
In order to evaluate the performance of SentiWS the dataset for evaluation was compiled manually. 2.000 sentences from a variety of Internet forums were manually categorized as being positive, negative, and neutral. Then selected 480 sentences (160 sentences of each category) to form the final data set for evaluation. Two persons annotated the selected sentences regarding the prior polarities of each adjective, adverb, noun, or verb in it. Then the words in sentences were tagged with POS, the raters' annotations were compared with entries in SentiWS and precision, recall, and F-measure were calculated. The overall (for positive, negative, and neutral sentences) evaluation

performance for SentiWS is showing good results with precision = 96%, recall = 74%, and F-score = 84% .

2.5.2 GermanPolarityClues

GermanPolarityClues is another publicly available German language resource for sentiment analysis. It consists of 10.141 polarity features, which have one of three polarities i.e. positive, negative, or neutral. GermanPolarityClues was built by semi-automatic translation of existing English sentiment analysis resources into German language. The translation was done of the polarity features only and the German terms received the weights of the corresponding English polarity terms. 290 German negation phrases were added to the feature collection and extended with new frequent positive and negative German language features. The evaluation results showed that the performance in metrics of F-score equal to 87.6% (Waltinger, 2010).

We have so far described the basic background information as well as presented different lexicon generation approaches and German-language resources for sentiment analysis, which may be used as a basis for Palladian Sentiment Classifier functionality. Now we will consider the machine learning techniques and particularly supervised learning, as this is the bases for Palladian Text Classifier performance.

2.6 Machine learning

Machine learning is used for automatic classification of the documents. There exist two main types of machine learning: supervised and unsupervised learning. The difference between them is that in the former the class labels (e.g. positive, negative, or neutral) are present in the data set before learning while in the latter the class labels are not provided that is why it is the task of the learning algorithm to analyse the internal document's (sentence's) structure and to assign class labels to them. The objective of the supervised learning is to create mapping (model) between the documents (sentences) and the class labels. Unsupervised learning is aimed at finding the intrinsic structure in the documents and to organize documents into the similarity groups (clusters) according to these common structures (Bing Liu, 2007).

2.7 Supervised learning

The document classification framework (supervised learning) has as a goal in the end to map the unlabeled documents (sentences) to their real classes. It usually consists of the following four steps (Markov, 2007):

1. *Data collection and preprocessing.* At this step the documents are collected, the document (sentence) classes are labelled, the features (e.g. n-grams, POS) are identified and the vector space representation of documents (sentences) is created. The collected data can be divided into two main subsets: *training set*, which is used for creating the model and the *test set* which is used for testing the model. Sometimes the training set is split into two subsets: the actual model construction subset and a model validation subset, which can be employed for tuning the learner parameters.

The following standard preprocessing techniques are used:

- Words splitting

Words splitting or tokenizing is decomposing the sentences into words.

- POS tagging

POS tagging assigns to every single word a label which correspond to its part of speech e.g. noun, adjective, verb, adverb etc.

- Stopwords removing

Not all words in the sentences carry useful information for classification task and that is why it is beneficial to get rid of such useless words. For German language some of such words are *das, die, der, aber, als, am, an, auch, auf, aus, bei, etc.* Removing these stopwords make the classification routine easier.

- Stemming

Stemming mainly deals with removing suffixes and prefixes from the words. The procedure of stemming can be explained by two sentences below. *I like to watch this movie.* And the second *I liked her performance, it was marvellous.* Here words *like* and *liked* would be treated by classification algorithm as two completely different words. That is why it is beneficial to stem these words to one common word *liki.* The stemming algorithm for German language is publicly available and will be used in this work.

- Feature selection

Features are the indicator of sentiment in the sentences. Often considering every single word in a sentence cannot be as indicative of positive or negative sentiment as when considering higher-level n grams. Let us take into consideration the following positive sentence *Dieser Film ist unheimlich schön.* When training (testing) the classifier and considering as features unigrams (every single word in a sentence) the word *unheimlich* would be considered as having negative tonality but when two consecutive words (bigrams) *unheimlich schön* are considered as a feature, the classification would produce correct results as these two words in combination result in a positive tonality. In this example word 2-gramms would perform better.

2. *Building the model.* At this step actual learning (training of the classifier) is done. It is usually the iterative and interactive process which is aimed at receiving the best *model* in the end. It includes the feature selection, *learning algorithm* application, and if needed tuning the learning algorithm.
3. *Testing and evaluating the model.* At this step the model is applied to the documents from the test set and their actual class labels are compared to the predicted ones.
4. *Classification of the new documents.* Using the model to classify the unlabeled documents (Markov, 2007). The steps of supervised learning are visualised in Figure 2.1.

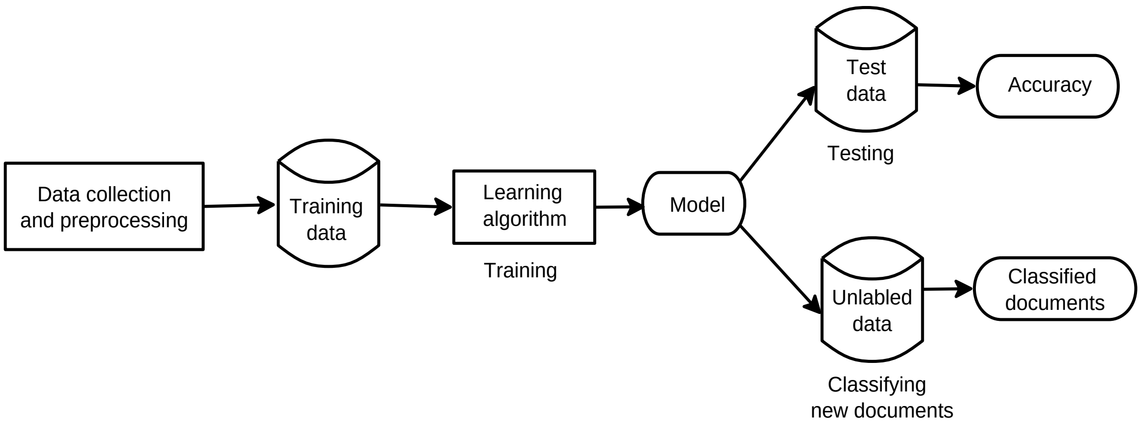


Figure 2.1: Steps of Supervised learning (Modified from Bing Liu, 2007, Figure 3.1.).

There are numerous effective supervised learning systems. Few of them are briefly described below.

2.7.1 Decision Tree

It is a very efficient classification algorithm in which the learned classification model is represented as a decision tree. It is beneficial to have a small tree as it tends to be more accurate and is easier to understand by human users. The tree only covers a subset of rules that exist in data, which is sufficient for classification. A decision tree partitions the training data set into disjoint subsets so that each subset is as pure as possible (contains training examples of a single class). Divide-and-conquer strategy is used for learning of a tree, it recursively partitions the data to produce the tree. The best attribute to partition the data at the current node is chosen with the aim to maximise the purity (Bing Liu, 2007).

2.7.2 Naive Bayesian Text Classification

It is the probabilistic approach to the text classification. Here the class labels are known and the goal is to create probabilistic models, which can be used to classify new texts. It is specifically formulated for text and makes use of text specific characteristics. The Naive Bayesian classifier treats each document as a ,,bag'' of words and the generative model makes the following assumptions: firstly, words of a document are generated independently of context, and, secondly, the probability of the word is independent of its position. This is why the name naive was used for this algorithm. In real text documents the words often correlate with each other and the position of the word in text may play role. The detailed description of this algorithm can be found here (Markov, 2007; Biu Ling,2007).

2.7.3 Support Vector Machines (SVM)

SVM is one of the most popular classification algorithms. It performs very accurate classification in many applications especially those involving high dimensional data. It is also one of the most accurate algorithms for text classification. In general SVM is a linear learning system that builds two-class classifiers. This algorithm finds the maximal margin decision boundary to separate positive and negative examples. Learning is formulated here as quadratic optimisation problem. It has also a solution for finding the nonlinear decision boundaries; to do this, the original data is transformed to the much higher dimensional feature space. But it has the limitation as it allows only two classes, i.e. binary classification. For multiple classification additional strategies should be applied (Bing Liu, 2007).

2.7.4 K-nearest Neighbour Text Classifier (kNN)

The idea of kNN is simple and quite effective in many applications e.g. text classification. Here, no model is learned from the data, learning only occurs when a test example need to be classified. Let *D* be the training data set. Nothing is done on the training examples that are why the algorithm sometimes is called lazy learning. When a test instance *d* is present, the algorithm compares *d* with every training example in *D* and computes the similarity or distance function between them. Every nearest neighbour (usually 3 or 5) votes with its class and the class of majority is assigned to the testing instance. The different values of *k* are tried and the *k* that shows the best performance on cross validation is selected (Bing Liu, 2007).

2.7.5 Dictionary-Based Classifier

This algorithm learns how probable each feature (e.g. n-gram) is for each given category (e.g. positive, negative, or neutral) and assigns the most probable category to the input document or sentence, depending on which level (document or sentence) the sentiment classification is done. A dictionary is built at a training stage by counting and normalizing the co-occurrences of one n-gram and a category. For the sentiment classification the dictionary might look as shown in the Table 2.2, where each column is a category and each row represents one n-gram. In each cell there is a learned relevance for each n-gram and a category - *relevance (n-gram, category)*. The sum of the relevance in each row must add up to one (David Urbansky, 2011).

For example the n-gram “glücklich” is more probable to be classified into “positive” category *(relevance(glücklich, positive) = 0.8)* than into “negative*” category (relevance(glücklich, positive) = 0.05).* But the n-gram “schlecht” is more probable to get the category “negative” *(relevance(schlecht, negative) = 0.7)* than “positive” *(relevance(schlecht, positive)) = 0.1)* etc.

n-gram	positive	negative	neutral
glücklich	0.8	0.05	0.15
schlecht	0.2	0.7	0.1
groß	0.3	0.2	0.5

Table 2.2: N-gram dictionary with relevancies for categories (modified version of Urbanksy, 2011, Table 5.1.)

In order to classify a new document (sentence) into one of the three categories, all n-grams must be created, afterwards the relevance scores (how probable is the n-gram for the category) should be taken from the dictionary and the category with the highest probability will be assigned to the document or sentence (David Urbansky, 2011).

The probability for each document (sentence) to be classified into one of the tree categories is calculated as shown in Equation 9, where $N_{sentence}$ is the set of n-grams for the given document or sentence.

$$CategoryProbability(category, sentence) = \sum_{n \in N_{sentence}} relevance(n, category) \quad (9)$$

If most of the n-grams in the document or sentence belong to the “positive” category, the document or sentence is going to get the “positive” category, otherwise “negative” or “neutral”.

2.8 Unsupervised learning

As was discussed below, the unsupervised learning algorithms try to find some intrinsic structure in data and to organise the documents into clusters. Two types of clustering are used: partitional and hierarchical clustering. K-means clustering algorithm is used in partitional clustering. Clustering utilises similarity function or distance function in order to measure how similar two objects are or to measure a distance between two data points. A document is represented usually as a “bag” of words in document clustering. A document can be represented as a vector and usually the cosine similarity function is used in order to compute the similarity between two documents (Bing Liu, 2007).

2.8.1 Supervised vs. Unsupervised learning for sentiment classification

A lot of researchers tend to use the supervised leaning techniques for the sentiment analysis tasks. SVM (see Section 2.9.6 and Section 2.9.7; as well as Dave, 2003; Pang, 2002), Naive Bayesian text Classification (Dave, 2003; Pang, 2002), kNN (Bing Liu, 2007), Dictionary-Based Classifier (Urbansky, 2011) were used for finding and classifying sentiments of the documents or sentences.

Unsupervised learning techniques are dealing with clustering the data set which is a challenging task because each clustering algorithm has limitations and works well with only certain data distributions. In the natural language processing it might be difficult to know what distribution the application data follows. Another challenge is how to standardize the data, to choose a suitable distance function (e.g. k in the k-means algorithms), which are also challenging tasks (Bing Liu, 2007).
That is why in this master thesis it was decided to use the supervised learning approach for learning Palladian Text Classifier.
In the following section the state-of-the-art opinion mining systems and the sentiment classification algorithms applied in them are outlined.

2.9 State-of-the-art analysis

This section reviews the state-of-the-art of opinion mining systems. By this analysis we want to point out on one hand the core characteristics of existing systems and on the other hand the central drawbacks they have in regards of sentiment classification for the German language. Eight academic opinion mining systems with main attention on the opinion and tonality classification algorithms applied in them are being analysed. The summary overview of the academic systems is presented in Table 2.3. Six commercial systems are being analysed as well and the summary overview is presented in Table 2.5. The main emphasis in analysis is made on the opinion classification task and on the opinion polarity identification algorithms.
For several years, a high number of frameworks for analysing human sentiments have been developed. Even though a lot of them have some common characteristics they differ in combinations of several features: sentiment classification algorithm, sentence scoring level, considered sources of information, sentiment tonalities, language to be analysed etc. (Bas Heerschop, 2010). Bas Heerschop (2010) investigated several existing approaches for sentiment analysis and suggested a scheme for their classification. The authors presented their own framework and researched the influence of negation on the sentiment analysis performance.
In this thesis the following features for state-of-the-art analysis and distinguishing the existing opinion classification frameworks are considered which were also mentioned in (Bas Heerschop, 2010):

Algorithm: A sequential set of steps which result in classification of a document (sentence) into one of the target categories.

Sentiment granularity level (SGL): Means on which level the sentiment is analysed: document, sentence, or window level.

Features: For different approaches different sets of features can be beneficial for the classification task (word-level n-grams, character-level n-grams, POS, the frequency of words in the document, presence of special signs e.g. ":-)", "!!!").

Document type (DT): Different types of documents require different techniques for opinion extraction and classification. Such types of documents are: web blogs, product reviews, twitter, regular text documents etc.

Sentiment type (ST): Different approaches investigate different opinion types (positive, negative, mixed, or neutral) expressed on the searched term.

Language: In the state of the art the frameworks intended for analysing German and English text sources are considered.

Following (Bing Liu, 2007) there are three main tasks which are addressed by researches dealing with opinion mining. They are: *sentiment classification, feature-bases opinion mining and summarization,* and *comparative sentence and relation mining*.

Sentiment classification treats the opinion mining task as a text classification problem. Opinion mining is usually done at the document or sentence level and the opinions expressed in documents (sentences) are classified as being positive, negative, or neutral. It does not consider the particular features (what exactly a person likes or dislikes). It discovers the general tonality of the opinionated sentence (Bing Liu, 2007). In *feature-based opinion mining and summarization* task the research is conducted in order to define the details i.e. what exactly a person likes or dislikes about the object (service, product etc.). Here, it is important to define the features which were commented and in some applications it is also important to define the opinion holder. Very often one object is assessed in comparison to another object. The goal of *comparative sentence and relation mining* task is to extract these comparative relations in sentences (Bing Liu, 2007).

In this thesis the focus will be directed to the algorithms satisfying the first (*sentiment classification*) task, though some algorithms for the *feature-based opinion mining* task are discussed as well. Bing Liu (2007) provided an overview of several algorithms for the sentiment classification task. They are: Classification based on sentiment phrases (see Section 2.9.3.), classification using a scoring function (see Section 2.9.1 and Section 2.9.2) and classification using text classification methods (employing any text classification algorithm e.g. naive Bayesian, kNN, or SVM see Section 2.9.6). Numerous other algorithms for sentiment classification do exist and several of them along with mentioned above three algorithms are going to be presented in the overview of the academic opinion mining systems. As the next step eight academic and six commercial systems are going to be presented accordingly.

2.9.1 Kushal Dave, 2003

The work of Kushal suggests a two-step algorithm for classifying product reviews. The algorithm is based on using a term scoring function. In the first step each term in the training set is scored, using the Equation 10.

$$score(f_i) = \frac{p(f_i|C) - p(f_i|C')}{p(f_i|C) + p(f_i|C')} \qquad (10)$$

Where f_i is a term and *C* is a class, *C'* is another class and $p(f_i|C)$ is the normalized term frequency which is determined by taking the number of times a feature f_i occurs in C and deviding it by the total number of tokens in *C*. Thus a term's score is the measure of bias towards either class ranging from -1 to 1.

In the second step the classification of the document into one of the classes (*C* or *C'*) is done by summing up scores of all terms. The sign of this sum determines the class of the document (*C* or *C'*). If document $d_i = f_{1 \dots} f_n$ then the document's class can be determined as shown in the equation,

$$class(d_i) = \begin{cases} C \ if \ eval(d_i) > 0 \\ C' \ if \ eval(d_i) \leq 0 \end{cases} \quad (11)$$

where

$$eval(d_i) = \sum_j score(f_j) \quad (12)$$

Experiments were conducted on the large number of English language reviews (more then 13.000) of seven types of products. The results showed that bigrams and trigrams as terms gave similar accuracies of 84.6%-88.3% on two different review datasets. Authors also experimented with some linguistic modifications using stemming, negation and collocations but it was not useful and on the contrary reduced the classification accuracy (Kushal Dave, 2003; Bing Liu, 2007).

The positive aspect of the suggested algorithm is that it is easily implementable. Thus, the disadvantage of this algorithm is that it classifies documents only into one of two classes positive or negative and was tested only on the English language corpus.
The classification algorithms in this master thesis are intended for categorizing German documents into three categories, namely positive, negative, and neutral. That is why provided by Kushal approach would not satisfy the research goals defined in this master thesis.

2.9.2 Bas Heerschop, 2010

The authors suggest a very simple sentiment classification framework which consists of algorithms for wordbank creation and lexicon-based document scoring. Only adjectives were considered for the wordbank as they are good sentiment indicators (Esuli, 2007). The authors also investigate how the recognition of negations influences the overall sentiment recognition performance. The goal of the classification framework is to classify document's sentiment, defining whether it has positive, neutral, or negative tonality.

Their wordbank contains per-word (adjective) sentiment score. The sentiment score of every adjective is calculated as following:

$$score(w) = \frac{\sum_{d \epsilon D_w} score\ (d) * \inf(\mathrm{w,d,neg})}{|D_w|}, \qquad (13)$$

where *score(d)* is manually assigned score for document *d*, $|D_w|$ is the number of documents in D_w and *inf(w,d,neg)* is relative influence of adjective *w* in document *d,* with *neg* indication accounting for negation or not. The relative influence is calculated as the count frequency *freq(w,d,neg)* of adjective *w* in *d* in relation to the total frequency $\sum_{w' \in d} freq(w', d, neg)$ of opinion carrying words *w'* in *d.*

$$\inf(w, d, neg) = \frac{freq\ (w,d,neg)}{\sum_{w' \in d} freq\ (w', d, neg)} \qquad (14)$$

Algorithm. Three algorithms were suggested: first for creating a list of adjectives (CLA) from the documents set, second for developing a wordbank (DW) (assigning scores to the adjectives), and third for scoring a document (SD) accounting for negations. The details of the algorithms can be found in paper and the final document score is calculated by Equation 15.

$$eval(d) = \sum_{w_i \epsilon d} (-1)^{negated\ (w_i,d)} * score(w_i)\ , \quad (15)$$

where $negated(w_i, d)$ is Boolean and indicates if the *i-th* adjective in *w* is negated in *d* or not, getting the value of 1 or 0 respectively. And depending on the value of the document score the document class (positive, negative or neutral) is determined. If the documents score is located between the values of –0.021 and 0.021 the document is considered to be neutral. If the score is higher than 0.021 the document is positive and if lower than -0.021 the document is classified as negative (Heerschop, 2010).

Result. The suggested system was evaluated against human's ratings. Two frameworks were built (one accounting for negation and another not). The former showed precision 71.23% and the latter 70.41%. It indicates that even though the negated sentences constitute only 0.85% in the original corpus the improvement of precision by 0.82% is considerable. The precision improvement is even better by 2.23% when applying the framework to the subset of the corpus in which all the documents contain a negated word.

The positive aspect is that in this system the authors account for negations but the drawback of this approach is that it only considers adjectives, but it was proven that for English language the combination of adjectives with other POS produces better classification results (Benamara, 2007).
That is why in PSC classification algorithm it will analysed to which extend considering all POS of German language influence the sentiment classification results.

2.9.3 Turney, 2002

The algorithm for sentiment classification suggested by Turney consists of three steps. The main goal is to classify the product's reviews as being positive or negative. The general idea is to extract phrases from the reviews which contain adjectives and adverbs, then to estimate the semantic orientation of these phrases and finally to determine the semantic orientation of the complete review (whether it is positive or negative). That is why the classification algorithm is called *Classification Based on Sentiment Phrases (CBSP)*

So the first step is extracting phrases containing adverbs and adjectives. For this purpose the POS tagging is implemented and the phrases corresponding to one of the predefined patterns are extracted. Here is an example of such a pattern. The first word from two consecutive words must be an adverb, the second word must be an adjective, but the third word cannot be a noun. If any two consecutive words meat this pattern then they will be extracted by the algorithm.

The second step is to estimate the semantic orientation of the extracted phrases, applying the Pointwise Mutual Information measure.

$$PMI\ (term_1, term_2) = \ log_2(\frac{Pr\,(term_{\ 1}\&term_{\ 2})}{Pr\,(term_{\ 1}) * Pr\,(term_{\ 2})}) \quad (16)$$

Where $Pr(term_1\&term_2)$ is the co-occurrence probability of $term_1$ and $term_2$ and $Pr(term_1) * Pr(term_2)$ is the probability that two terms co-occur if they are statistically independent. The ratio of the probabilities is the measure of degree of statistical dependence of two terms, when their *log* is the amount of information we acquire about the presence of one of the words when we observe another (Turney, 2002).

The semantic orientation (SO) of the extracted phrases is calculated based on its association with the positive reference word “excellent” and the negative reference word “poor”.

$$SO\ (phrase) = PMI\ (phrase, \text{“excellent”}) - PMI\ (phrase, \text{“poor”}) \quad (17)$$

The probabilities are calculated by issuing queries to the search engine and collecting the number of hits. Here, the number of relevant document to the query is the number of hits. The author used the search engine AltaVista which has a NEAR operator that constrains the search to documents that contain the words within ten words of one another, in either order (Turney, 2002).

Let *hits(query)* be the number of queries returned, then estimate of SO using Equation 16 and Equation 17 can be rewritten as following if the co-occurrence is interpreted as NEAR

$$SO(phrase) = log_2(\frac{hits(phrase\ \ NEAR\ \text{excellent})hits("poor\ ")}{hits(phrase\ \ NEAR\ \text{poor})hits("excellent\ \ ")}) \quad (18)$$

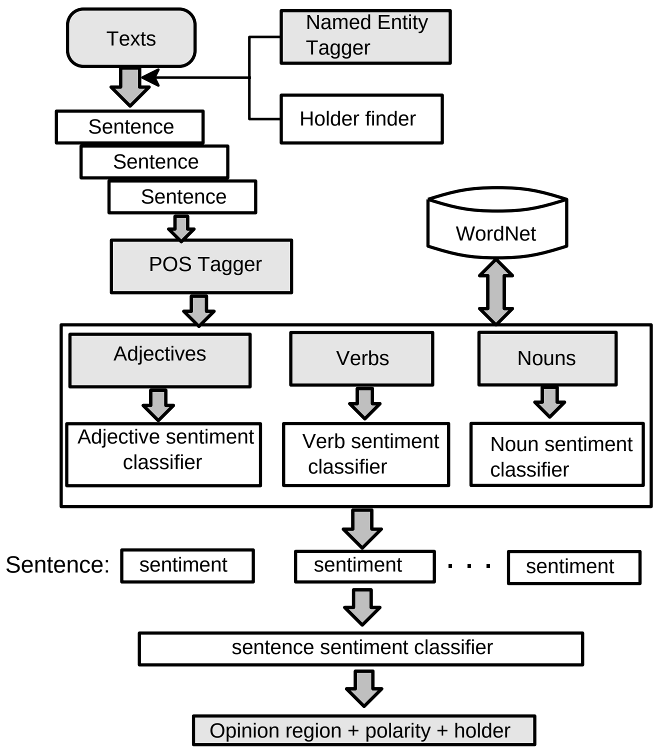
In the final third step the algorithm computes the average SO of all extracted phrases in review and classifies review as recommended if average SO is positive and as not recommended if average SO is negative.

Results. The classification accuracies on reviews from various domains show different results from 84% for automobile reviews to 66% for movie reviews.

The drawback of algorithm is that it can classify sentences only into two categories, namely positive or negative and it does not consider negations. But it was analyzed (Heerschop, 2010) that considering negation improves the classification performance. For example the SO of the phrase can be defined by this algorithm as positive, but the adjective or adverb can be negated in the sentence, resulting in negative tonality and as a result the algorithm would produce the wrong results.

2.9.4 Soo-Min Kim, 2004

The goal of the suggested system is to find a sentiment holder and to define the type of sentiment which person or organisation expresses on a given query topic. They developed a four step algorithm for sentiment analyses: firstly, sentences that include topic phrase and information holder are extracted, then a holder-based region of opinion (complete sentence or sentence region, called window) is defined. Then the sentence sentiment classifier defines the polarity of all sentiment words and the last step is to combine them to get the final result i.e. holder’s opinion on the topic. General system architecture can be seen in Figure 2.2.



Texts
Named Entity Tagger
Holder finder
Sentence
Sentence
Sentence
POS Tagger
WordNet
Adjectives
Verbs
Nouns
Adjective sentiment classifier
Verb sentiment classifier
Noun sentiment classifier
Sentence:
sentiment
sentiment
. . .
sentiment
sentence sentiment classifier
Opinion region + polarity + holder

Figure 2.2: General system architecture (Kim, 2004).

Classification part consists of two parts, namely word sentiment classifier and sentence sentiment classifier. In *word sentiment classifier (WSC)* a wordbank is created in the following way. First the seed set of words is selected and then these words are extended with their synonyms from the WordNet. In order to omit the problem of ambiguous words (words that may have both positive and negative tonalities depending on the context), the technique for weighting the strength of sentiment polarity is applied by solving the Equation 19 (Soo-Min Kim, 2004).

$$\arg\max_c P(c|w) = \arg\max_c P(c|syn_1, syn_2 \dots syn_n) \qquad (19)$$

Where *c* is the sentiment category (positive or negative), *w* is an unseen word and syn_n are the WordNet synonyms for *w*.

In *Sentence sentiment classifier (SSC)* in order to perform sentence sentiment analysis the holder of the sentiment is defined and a region near the holder in which the opinion can be expressed. Three different models for assigning a sentiment to the sentence are suggested. The first model is simple and only considers the polarities of the sentiment; the second/third model takes into consideration the sentiment strength and represents the harmonic/geometric mean of the sentiment strength in the region. It means if a region contains more and stronger positive than negative words, the sentiment is considered to be positive (Soo-Min Kim, 2004). Four types of regions (windows) are considered: a complete sentence, words between a holder and topic, second variant to the end of sentence and second variant plus/minus two words (Soo-Min Kim, 2004).

Results. The best overall performance was provided by the first model with accuracy 67% for automatic opinion holder detection.

This system is focused on defining the opinion holder, which is not the goal in this master thesis, though the suggested sentence classification models are easily implementable and could be used by other researchers for similar tasks (the identification of opinion holder and defining the sentiment of expressed opinion).

2.9.5 Carmine Cesarano, 2006

Cesarano suggests a set of algorithms for evaluating the degree of sentiment expressed on target topics in the documents. The system provides both quantitative (larger the number, more positive the opinion) and qualitative opinion (e.g. harsh, complimentary) results. The authors claim that their work is different from the state-of-the-art in the following aspects:

- The opinion scores for the documents are continuous (expressing the degree) rather than binary;
- Developed multiple scoring methods, including qualitative scoring method;
- Developed a model to combine multiple scoring methods together.

Opinion analysis architecture. The developed system architecture consists of the next components: user specification (e.g. URLs), web spider, scoring opinion expressing wordbank, quantitative opinion analysis algorithm, and qualitative scoring model (Cesarano, 2006).

The authors used the supervised method of wordbank creation. In result every word is assigned a sentiment score associated with it. The authors suggested two methods for scoring sentiment bearing words: pseudo-expected value word scoring and pseudo standard-deviation adjective scoring which got the idea from the concept of expected values in statistic (Ross, 2001). The authors assert and proved that the effective sentiment analyses rely on accurate word scoring as well as accurate document scoring (Cesarano, 2006).

The pseudo-expected word scoring is calculated by Equation 20.

$$pevs^k(w) = \frac{\sum_{d \in D_{test}} (avsc^k(d) * \frac{n(w,d)}{\sum_{w' \in oew(D_{test})} n(w',d)})}{\sum_{d \in D_{test}} avsc^k(d)} \qquad (20)$$

Where $avsc^k(d)$ is the averaged score manually assigned to the document by human users, not taking into account *k* top and *k* bottom scores in order to eliminate outliers.

n (w,d) is a number of concurrencies of word w or its synonyms in document d,

D_{test} is a set of test documents,

oew (d) – a set of all opinion-expressing words (and their synonyms) occurring in document d

oew(D_{test}) - a set of all opinion-expressing words (and their synonyms) occurring in a set of test documents D_{test} .

The score $pevs^k(w)$ of word *w* is calculated by averaging the contribution of the score of *w* across all the documents in the test set.

Document opinion scoring algorithms. Four document scoring algorithms which in result define the sentiment of document (sentences) have been suggested:

- Topic-Focused (TF) algorithm
- Distance-weighted topic focused (DWTF) algorithm
- Template-based (TB) algorithm
- Hybrid Evaluation Method (HEM)

The idea of *Topic-Focused algorithm* is to find all the sentences that include sentiment on the target topic, calculate their sentiment score and return the average sentiment score for the document. The sentence is scored based on the score of every opinionated word which is present in this sentence. *Distance-weighted topic focused algorithm* is window-based approach, while it divides all the sentences of a document into those that contain a topic keywords and those that don't and calculates the sentiment score of all the sentences, giving more weight to the sentiment words that are near the topic keywords. *Template-based algorithm* only considers the sentences that match predefined templates. It uses the same technique as TF algorithm to assign the score to the document's sentences. *Hybrid Evaluation Method* works as following: it associates a vector with each document *d*. This vector consists of functions to assign the scores to document (e.g. three methods described above could be used for this purpose). The HEM algorithm looks at the k-nearest neighbours for the document associated vector and calculates the document score as an average of scores assigned to the neighbours by human which were evaluating the documents (Cesarano, 2006).

Results. The system was trained with 352 news articles on 12 topics. At the evaluation phase the system showed good results on extracting sentiment from weblogs with 91% precision (73% recall) in detecting "positive" weblogs, 78% precision (91% recall) for "negative" weblogs and 55% precision (37% recall) for "very negative" and "very positive" weblogs .

Rather than classifying sentences into one of three tonalities, the authors concentrated on determining a degree to which the documents are positive or negative. They did not consider detection of neutral sentences, which would not satisfy the research goals of this master thesis. In order to establish a suggested mechanism of scoring the wordbank, the authors of the paper needed to assign scores to the set of English testing documents manually, which required 16 persons to establish a sufficient testing dataset and to produce results which are not biased towards a single person's estimation. This approach would be too work-intensive to realize it in the case of this work.

The next three opinion retrieval systems were presented at TREC conference (2007-2008). Different mechanisms for opinion retrieval and methods for opinion polarity identification are suggested. The opinions directed to the terms are extracted from the weblogs.

2.9.6 Wei Zhang, 2007

Wei Zhang presents a three-step algorithm for opinion retrieval task, which are*: information retrieval, opinion identification,* and *ranking step.* The general structure of an opinion retrieval system is shown in Figure 2.3.

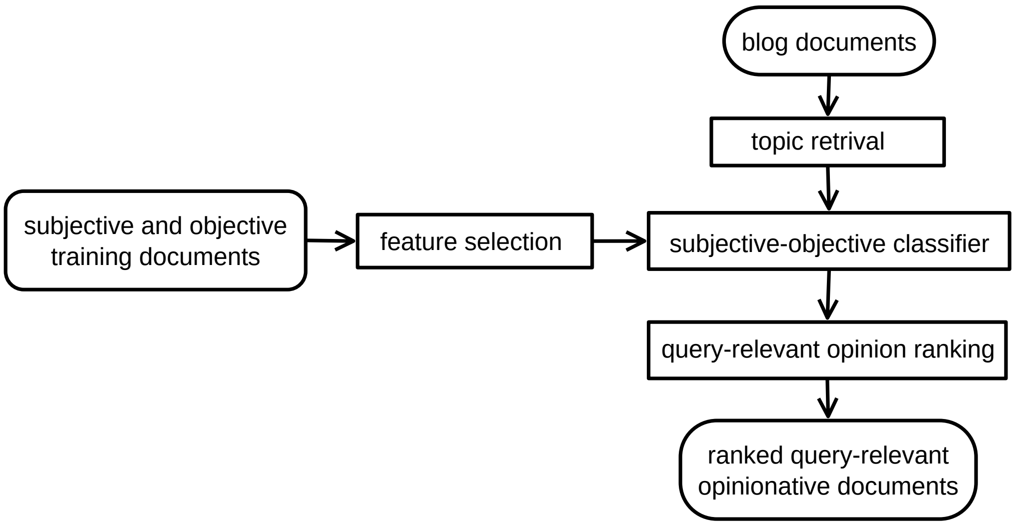

blog documents
topic retrival
subjective and objective training documents
feature selection
subjective-objective classifier
query-relevant opinion ranking
ranked query-relevant opinionative documents

Figure 2.3: The structure of the opinion retrieval system (Wei Zhang, 2007).

The goal of the system is to return blog posts that contain opinions and define the polarity of these opinions. In *information retrieval step* the documents relevant to the query topics are retrieved. The following mechanisms are applied for improving the retrieval effectiveness: concept identification, three query expansion methods (utilizing Wikipedia, pseudo feedback, and Web-based), phrase similarity calculation and document filtering techniques. WordNet and Wikipedia are used for proper noun and dictionary phrase identification (Wei Zhang, 2007).
For determining whether the retrieved document is subjective (contains opinion) or objective (contains factual information) a supervised learning method (SVM classifier) is used.
Opinion identification step is needed for finding the opinionated text in document. Due to the fact that in text document feature space is too big (e.g. all terms in the document), the amount of features could be reduced by applying Chi-square method for feature selection. The benefit of reducing the number of features is shorter execution time and better performance (Phayung Meesad, 2011). Chi-square method measures the lack of independence between the terms in the category (Saengsiri, 2010).
Chi-square test is applied to objective and subjective training data in order to select features. These features are then used for building a support vector machine classifier. The subjective training data is obtained from review sites e.g. retail.com. Then the built classifier is applied to all the documents. The documents are split into sentences and the classifier tests all of them and as a result the sentences receive either a subjective label (with its strength) or an objective label. The document is said to be opinionated if it contains at least 1 subjective sentence (Wei Zhang, 2007).

Ranking step is implemented using the text window method (five sentence window within which the original or expanded query term has to be present). It is done in order to be sure that the opinion in the document is directed to the query terms.
For defining the polarity of the documents authors use the supervised learning method (SVM classifier) as well.

In *polarity task* the positive, negative, or mixed label is assigned to the documents. Two opinion retrieval systems (for retrieving positive and negative opinions) are constructed and the judging function is used for assigning a mixed label to those documents which have some defined level of sentences with both opinions. The structure of polarity classification system is shown in Figure 2.4.

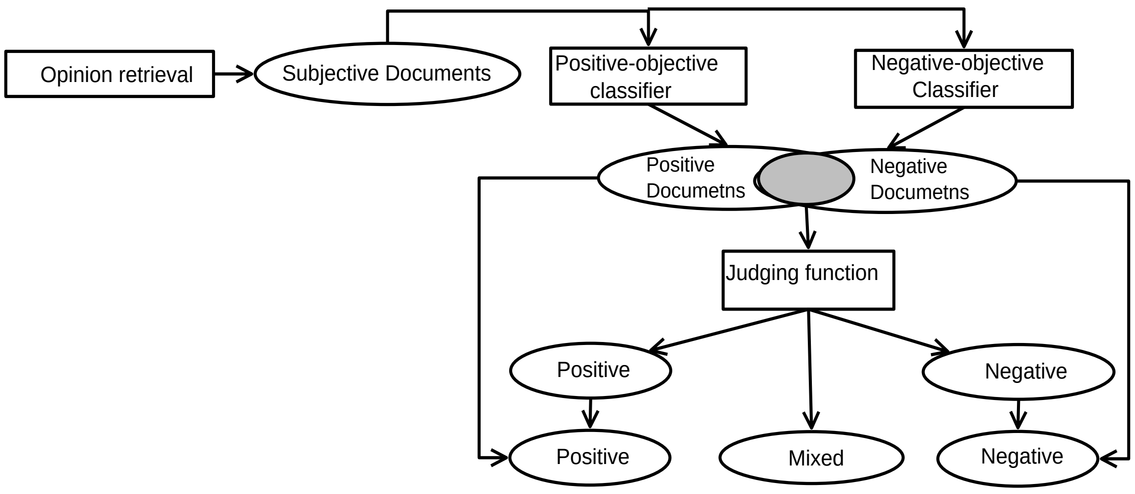


Figure 2.4: The structure of the polarity classification system (Wei Zhang, 2007).

To train the tonality classifier the training data is collected from a number of review sites (e.g. retail.com) with the review ratings. The high rating scores are considered as positive opinion data while the low as negative opinion. The classifier trained with positive and negative reviews is used to determine the polarity of the document (positive, negative, or mixed).

Results. The main task of the system was to test the classification performance when varying two parameters. These parameters are the weights of the expanded query terms and the opinion similarity functions. The four runs were submitted for opinion retrieval task and four runs for polarity classification task. The best mean average precision results for one of the system configurations for opinion retrieval task is equal to 0.43. For the polarity classification task the best result score of classification correctness is equal to 0.37 for the ten topmost documents.

This system suggests SVM, alternative supervised learning technique for documents classification. However, it does not define the sentences of neutral tonality but positive, negative, and mixed. While in this master thesis dictionary-based approach is chosen for learning PTC, which allows classifying sentences into positive, negative, and neutral tonalities.

2.9.7 Lifeng Jia, 2008

This opinion mining system is built based on the Wei Zhang system (see Section 2.9.6). Lifeng Jia defins two main modules - a four-step opinion retrieval module and the polarity classification module. The goal of the system is to retrieve blog documents that contain opinions directed to the query topic and to define the polarity of these documents.

The opinion retrieval is considered as the four step procedure:

- The information retrieval step (retrieving documents relevant to the query);
- Abbreviation identification (is a new step in comparison to Wei Zhang system). It serves to improve opinion identification.
- Opinion identification step (finding opinionated text in documents).
- Ranking step (the documents are ranked using the information retrieval (IR) score and the opinionated score of each document).

As in the system discussed above here the supervised learning algorithm in case study of SVM classifier is used in order to find the subjective documents and another two SVM classifiers are used to determine the polarity of the documents. The NEAR operator is used to determine whether the opinion, expressed in the document, refers to the query terms. Both sentence level opinion polarity classification and based on it document level opinion polarity classification are implemented in the polarity classification module. The general architecture of the opinion classification system is shown in Figure 2.5.

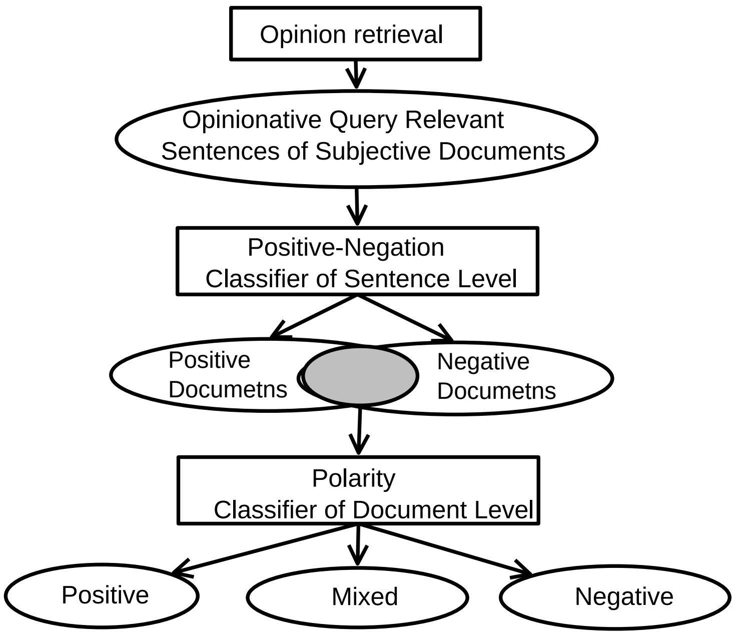

Opinion retrieval
Opinionative Query Relevant Sentences of Subjective Documents
Positive-Negation Classifier of Sentence Level
Positive Documetns
Negative Documetns
Polarity Classifier of Document Level
Positive
Mixed
Negative

Figure 2.5: The architecture of the polarity classification system (Lifeng Jia, 2008).

Each opinionated sentence that refers to the query terms receives by SVM classifier a polarity label and a confidence score. This information is then used for defining the overall document polarity. In this case the document is considered to be positive (negative) if it includes only positive (negative) query-relevant sentences and mixed if it includes the sufficient number of both positive and negative sentences (Lifeng Jia, 2008).

Results. The authors submitted 21 opinion runs based on six baselines (five baselines from the system described above and one their own baseline) to which they applied their opinion identification technologies. Each baseline consists of at most 1000 documents for each query which are ranked in descending order of IR scores. For opinion identification runs the best score for mean average precision is equal to 0.47 and for polarity classification task the R-Precision score of positive and negative ranking is equal to 0.22.

The system does not provide any mechanism for classifying documents into neutral category. Nevertheless, the suggested supervised learning approach could be seen as an alternative to the Dictionary-based learning approach suggested in this master thesis. Thus, the SVM classifier could be considered in the outlook and future research work when applying supervised learning techniques for sentiment analysis.

2.9.8 Breyten Ernsting, 2007

The suggested opinion retrieval and analysis approach consists of three steps: topic retrieval, opinion finding, and polarity identification. The goal of the work is to return blog posts that contain opinions and to derive their polarity e.g. positive, negative, or neutral. The authors argue that a strong topic retrieval system is the most important part of the opinion finding (Breyten Ernsting, 2007).

For the *topic retrieval task* the language models are used. The authors apply external expansion as well as query rewriting strategies in order to improve the retrieval efficiency.

Opinion finding methods are dealing with defining opinionated blog posts and are implemented using query independent document priors. They compare two document priors for being opinionated: a lexical approach (LA) and a comment based approach (CBA). For the lexical approach a list of opinionated words (only strong positive/negative words) from the OpinionFinder[3] system is collected (Breyten Ernsting, 2007). For lexicon-based approach document priors p(d) are estimated by Equation 21.

$$p(d) = \sum_{i=1}^{w} c(w_i, d) * |d|^{-1}, \quad (21)$$

where *w* is the list of opinionated words, $c(w_i, d)$ is the count of opinion word i in document d and |d| is the document length in words.

The second - comment based approach assumes that if a blog post is opinionated it has a lot of comments left by users and the document priors p(d) in this case are estimated by Equation 22.

$$p(d) = \log\left(N_{comments\ ,d}\right) \quad (22)$$

Where $N_{comments\ ,d}$ is the number of comments in document d.

For *polarity identification* two approaches are used. The first is based on the list of words from lexical approach and suggested mathematical equations to estimate the polarity of the blog post.

$$pol(d) = \begin{cases} positive\ if\ r(d) > 0.01 \\ negative\ if\ r(d) < -0.01 \\ neutral \qquad otherwise \end{cases} \quad (23)$$

Where r(d) is identified as

$$r(d) = \left(\sum_{i=1}^{n} c\,(n_i, d) - \sum_{i=1}^{p} c(p_i, d)\right) * \ |d|^{-1}, \quad (24)$$

Where n is the list of negative words and p the list of positive words, $c\,(n_i, d)$ is the number of times word n occurs in document d and |d| the document length in words (Breyten Ernsting, 2007).
For the second approach they consider punctuations. For instance exclamation marks, question marks, ellipsis and caps strings of more than three characters are considered as indicators of negative document tonality. Taking this consideration into account the document's polarity is estimated according to Equation 25.

$$pol(d) = \begin{cases} positive\ if\ r(d) < 0.1 \\ negative\ if\ r(d) \geq 0.1 \end{cases} \quad (25)$$

[3] URL: http://www.cs.pitt.edu/mpqa/

And ration for these punctuation indicators r(d) is defined as following.

$$r(d) = c(i,d) * |d|^{-1} \quad (26)$$

Where c(i,d) is the total number of occurrences of these special signs (punctuations).

Results. Seven different runs were suggested and the performance of runs on topic retrieval, opinion retrieval and polarity identification were investigated. The run with external query expansion technique showed the best performance for opinion retrieval task with precision at ten topmost documents p@10 equal 56%. It was discovered that lexicon-based approach as well as comment-based approach have positive influence on opinion retrieval but lexicon-based approach performs better. In regards of polarity identification of documents, the approach based on difference between negative word ration and positive word ration is only slightly better than the polarity identification approach based on considering punctuation marks (Breyten Ernsting, 2007).

The comment-based approach for opinion retrieval and corresponding polarity identification approach based on usage of punctuations are quite interesting mechanisms, though they are more suitable for highly opinionated Internet resources such as blog posts and might not suite for analyzing the sentiment of datasets considered in this thesis, which are derived from a variety of Internet resources. The drawback of the punctuation approach is furthermore, that it can classify the documents only into two polarities, namely positive or negative. The lexicon-based approach suggested by Ernsting utilizes English language resources, as it is intended for English language content classification.
In this thesis the lexicon-based PSC classifier applies a German language sentiment dictionary for analyzing German language content.

2.9.10 Summary of academic approaches

Eight academic opinion mining systems have been analysed so far considering their approaches and algorithms for opinion identification and determination of opinion polarity. The main emphasis in state-of-the-art was made on the *sentiment classification task* and its opinion classification algorithms. A number of opinion mining systems for classification task use supervised learning techniques (e.g. SVM, kNN etc.), perform classification using the scoring functions (see Section 2.9.1 and Section 2.9.2), or use the lexicon-based approach (see Section 2.9.2; 2.9.3; 2.9.8). For the latter it is often important to first develop the lexicon wordbank, which includes a per-word sentiment score. The wordbank can be created completely manually, using the supervised learning on the set of manually rated documents, or learning through related word expansion (Section 2.4; 2.9.2; 2.9.4; 2.9.5; 2.9.8). The latter is done by propagating the seed set of words to the available lexicon resources such as WordNet, SentiWordNet etc. and iteratively extending this set of words.

Even though numerous opinion mining systems have been analysed it is difficult to say which of applied classification approaches and algorithms perform best. All of these systems show more or less good results but only applying their own datasets. It would only be fair to say that one particular approach and classification algorithm perform best if it was applied and tested on the same dataset. The results of classification algorithm for one dataset might be good, but applying it to another dataset could show unsatisfactory results. That is why the performance of two classifiers considered in this master thesis, namely Palladian Text Classifier and Palladian Sentiment Classifier will be tested on the same dataset in order to define which of them performs better.

All observed in state-of-the-art systems are intended for classification of English language documents, but one for Dutch language, and the results of their classification cannot be compared with classification results of PTC and PSC classifiers, because they are used for analysing German language documents and were tested on German corpus.
Among the reviewed opinion mining system some of them are using machine learning techniques for classification tasks, others are applying lexicon-based approach. Those of the observed systems which employ lexicon-based approach in order to define document's polarity were utilizing powerful English-language resources like WordNet, but unfortunately such powerful tools are not yet available for German language. Though publicly available sentiment dictionaries for German language already exist and one of them will be applied in PSC classification algorithm.
Some of the observed systems and classification algorithms are capable of classifying documents only into two tonalities, namely positive and negative; others consider only selected set of features (e.g. adjectives). A lot of lexicon-based approaches do not consider negation in classification task. That is why classification algorithms, described in these approaches would not be able to be used for classification task in this master thesis. Developed in this master thesis preprocessing and classification algorithm for PSC is designed for classifying German documents, applying all POS as features and is able to classify sentence into three tonalities. As well as described supervised learning approaches would not satisfy the considered research question which will be attended by PTC.
The overview of observed academic opinion retrieval systems is presented on Table 2.3. The list of abbreviations used in this table is presented in Table 2.4.

System	Algorithm	SGL	Features	DT	ST	Language
Kushal, 2003	CSF	document	Bigrams, trigrams, negation	Product reviews	positive, negative	English
Heerschop, 2010	CLA,DW,SD	document	POS, n-grams, negation	Text documents	positive, negative, neutral	Dutch
Turney, 2002	CBSP	document	POS, n-grams	Product reviews	positive, negative	English
Soo-Min Kim, 2004	WSC, SSC	sentence, window	unigrams, bigrams, trigrams	Text documents	positive, negative, neutral	English
Cesarano, 2006	TF DWTF TB HEM	sentence window sentence sentence, window	POS, n-grams	News articles, Weblogs.	positive, negative, neutral with opinion degree	English
Wei Zhang, 2007	SVM	document	Unigrams, bigrams	Blog posts	positive, negative, mixed	English
Lifeng Jia, 2008	SVM	sentence, document	unigrams, bigrams	Blog posts	positive, negative, mixed	English
Ernsting, 2007	LA, CBA	document	Unigrams, expressive language (e.g.!!)	Blog posts	positive, negative, neutral	English

Table 2.3: Overview of opinion retrieval systems.

As it can be seen in the table, currently no approach exists that focuses on sentiment classification for the German language. This gap should be closed by this master thesis by analyzing concrete classification approaches and determining their characteristics if they are applied to the German language.

Abbreviation	Meaning
SGL	Sentiment granularity level
DT	Document type
ST	Sentiment type
CSF	Classifier using a Scoring Function
CLA	Algorithm for Creating a List of Adjectives
DW	Algorithm for Developing a Wordbank
SD	Algorithm for Scoring a Document
CBSP	Classification Based on Sentiment Phrases
WSC	Word Sentiment Classifier
SSC	Sentence Sentiment Classifier
TF	Topic-Focused Algorithm
DWTF	Distance-Weighted Topic Focused Algorithm
TB	Template-Based Algorithm
HEM	Hybrid Evaluation Method
SVM	Support Vector Machine
LA	Lexical Approach
CBA	Comment Based approach

Table 2.4: Abbreviations.

2.9.11 Commercial systems

After getting an overview of the existing opinion mining academic systems it is interesting to know which working opinion mining systems are currently available on the market. Even though some number of commercial systems do already exist, they all keep the implementation details secret e.g. which algorithms have been used for classifying the documents (sentences) as subjective, what are the algorithms for opinion’s polarity identification, etc.

Among the available working commercial systems the sentiment analysis services are provided by the following systems OpenSecurity (OpenSec), Mantoo, I-sieve Technologies, Hour1, Radian6, Trackur etc. Most of these systems provide customers with a dashboard where a customer can see how the users’ preferences and opinions about a certain product (services) change over time. For the companies which want their company’s reputation to be professionally managed the tools provided at reputationdefender.com could be used as well.

Some short summary of the studied opinion mining commercial systems and their websites are presented in the Table 2.5. Sticking to this master thesis the commercial opinion mining systems are characterized according to the following criterion: the languages of sentiment analysis, opinion polarities, whether the approach is fully automated or not, sources of information considered in sentiment analysis, and opinion mining approaches.

Opinion mining system / Link	Language	Opinion Polarities	Sources of information	Automatization	Approach
OpenSec http://www.opsecsecurity.com/	German, English.	positive and negative.	Online communities, Discussion boards, Weblogs, Product rating sites, Chartrooms, Price comparison portals, Newsgroups	Information not provided	Selected information sources are mined to retrieve useful quantitative and qualitative data. User content is aggregated, evaluated, and interpreted to provide analysis, insights, and recommendations.
Mantoo http://www.maanto.de/	German, English.	positive, negative	Blogs, forums, product reviews	Automatic analysis combined with manual quality control	Usage of the text mining techniques for automatic opinion analysis
I-sieve Technologies http://i-sieve.com/	English	positive, negative, neutral	Blogs, Twitter, Social Networks, Professional media, Discussion forums , YouTube comments	Automatic analysis combined with manual quality control	Application of video, audio and text recognition patterns for sentiment identification and classification.
Hour1 http://www.hour1.de/	German and English	positive, negative	Blogs, forums, product reviews, Social Networks e.g. Facebook, Twitter, Studivz, Xing	automatic	Usage of pattern-matching algorithm for comparing independent parts of text with already existing patterns.
Radian6 http://www.radian6.com/	German, English, French, Portuguese, Spanish	positive, negative	Blogs, Twitter, news sites, forums, videos	automatic	Information not provided
Trackur http://www.trackur.com/	English	positive, negative	Blogs, Twitter, Facebook, forums	automatic	Information not provided

Table 2.5: Summary of commercial sentiment analysis systems.

Thus, as we can see in the table the classification approaches are kept secret, or only very general idea is provided.

2.10 Summary

In this chapter the basic terms, concepts and algorithms used in opinion mining research area have been discussed. Different sentiment analysis features and opinion lexicon generation techniques have been analysed. Then the main evaluation measures which are used to assess the performance of the opinion mining system were introduced. Afterwards two new lexical resources for German language i.e. SentiWS and GermanPolarityClues have been presented. The introduction to two machine learning techniques supervised and unsupervised learning has been done and the state-of-the-art of opinion mining systems has been reviewed. The chapter closes by summary of reviewed opinion mining systems and by describing several commercial opinion mining systems.

The next chapter introduces the concepts of Palladian Text Classifier and Palladian Sentiment Classifier, describes the experimental setups of both classifiers and presents the implementation details of realizing PTC and PSC.

Chapter 3: Concept and implementation

As a preliminary step for the in-depth analysis of the classification behaviour of the selected classifiers and for addressing the research questions stated in Chapter 1, this chapter will present the created experiment setup, developed classification algorithms on conceptual level, and necessary software that has been developed and used in our research approach.

After an overview, the experiment workflow has been presented in Section 3.1, Section 3.2 present the concept of Sentimal, Section 3.3 introduces to the concept of PTC. The implementation details for realizing both Sentimal and PTC are described in Section 3.4. Section 3.5 presents a brief summary of this chapter.

3.1 Overview

To decompose our research approach into coarse-grained steps, three obvious steps can be identified: in a first step, different configurations for the classifier executions have to be defined. In a second step, a classification of predefined datasets will be carried out using the two selected classifiers and their configuration options. In a third step, a result analysis is realized. Figure 3.1 visualises these steps and associates central questions and aspects with them which will be discussed in the following.

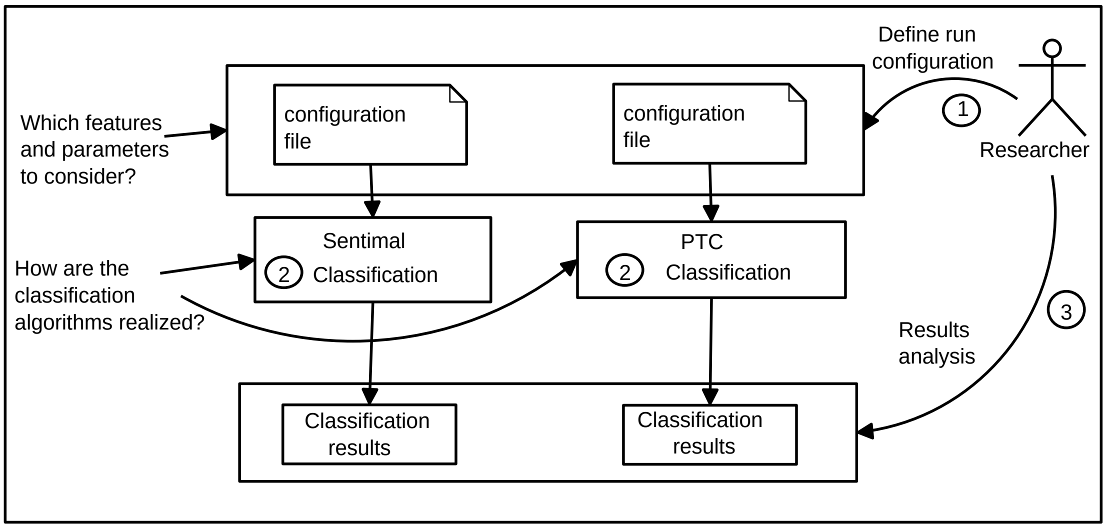


Figure 3.1: General structure of experiment workflow.

Which features and parameters should be used for each of the classifiers? How should the classifiers be realized? These questions will be addressed in the following.

Our research focuses on two classifiers, which are the Palladian Text Classifier (PTC) and Palladian Sentiment Classifier (PSC). The introduction to the principles of work of the former was provided in Section 2.7.5. The Palladian Sentiment Classifier will be called here and for the rest of the document **Sentimal** in order not to confuse it with PTC**.** Before details of these classifiers will be discussed in the Section 3.2 and Section 3.3, we will provide an overview of their characteristics in the following.

Even though both classifiers are dictionary-based, there is a significant difference in their characteristics. PTC classifier's performance is based on the supervised learning and thus the classifier first must be trained on a training dataset, and afterwards tested on another testing dataset in order to achieve the results of its classification. The dictionary in PTC is developed at the training step by counting and normalizing the co-occurrences of the words (n-grams) and a category.
Sentimal is a lexicon-based classifier. It utilizes in classification task already available resource for sentiment analysis and opinion mining which is called SentimentWortschatz (see Section 2.5.1). In SentiWS every opinion carrying word has the assigned polarity score, which is used by Sentimal's classification algorithm in order to define the final document polarity.
The general structure of performance steps can be seen in Figure 3.2.

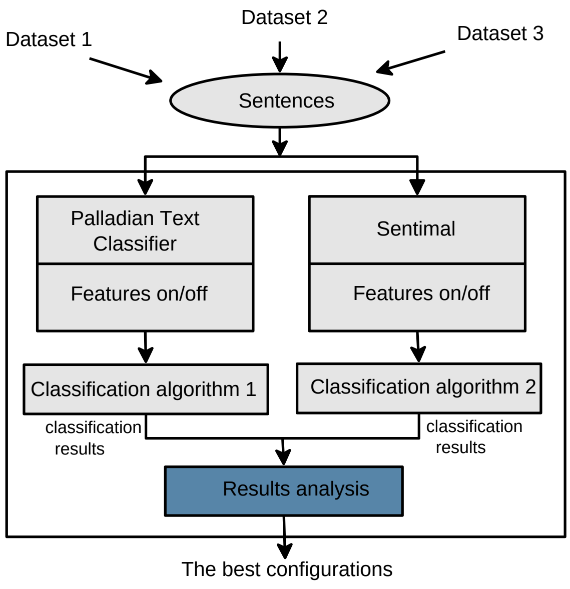


Figure 3.2: General structure of performance steps.

Two sentiment classifiers (PTC and Sentimal) receive sentences derived from different datasets as an input. These sentences are going to be processed by the classification algorithms. Classifiers are configured to work with a different set of features in order to define the best configurations that show the best performance results, which in the end is the main goal of the master thesis. The classification pipeline for both classifiers is visualized in Figure 3.3.

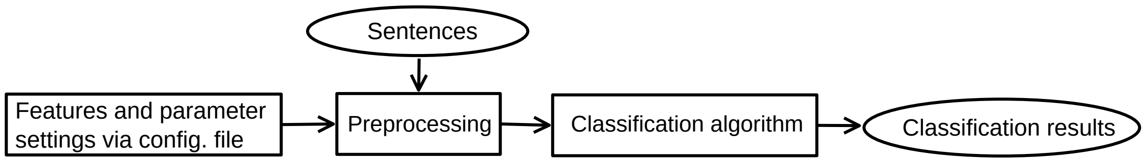


Figure 3.3: Overall classification pipeline.

In the following it will be described how all these aspects are addressed and realised in each of the two selected classifiers.

3.2 Sentimal

This section is structured as the following. First the experimental setup is presented in Subsection 3.2.1 and then the general preprocessing technique and applied classification algorithm is conceptually described in Subsection 3.2.3.

The overall structure of Sentimal is presented in Figure 3.4. In this figure the applied preprocessing steps, features and the general idea behind the classification algorithm, namely the usage of public resource for sentiment analysis is presented.

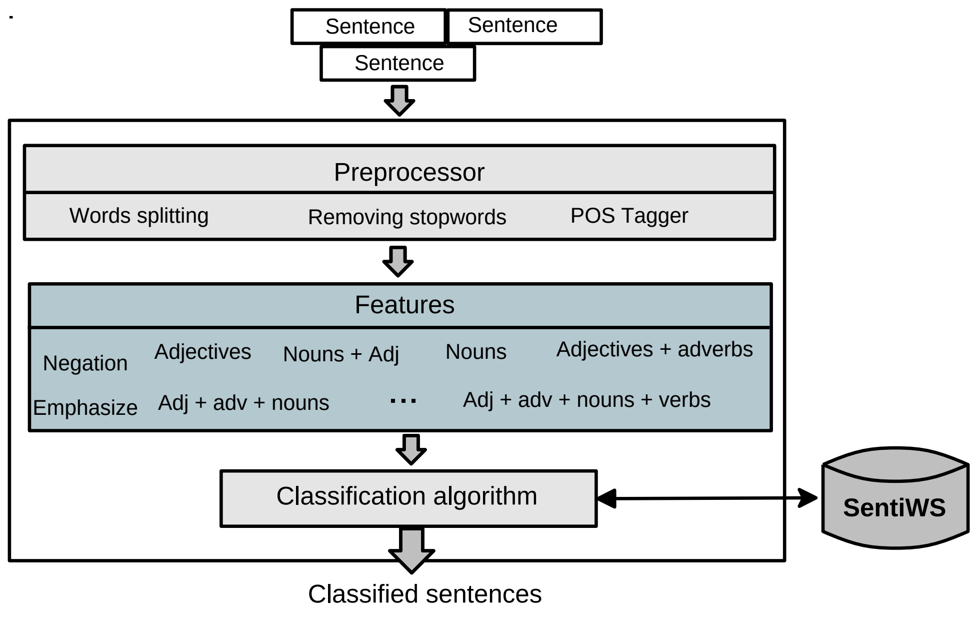


Figure 3.4: General structure of Sentimal.

The features and other settings which can be configured per every system run via configuration file will be described in the following subsection.

3.2.1 Sentimal experimental setup

First of all we are going to define a set of features for Sentimal. These features are indicators which help to determine if the sentence is opinionated or not. Due to the fact that Sentimal is a lexicon-based classifier, here it will be operated with different parts of speech as features. Sentimal classification algorithm is employing in its work SentiWS, which contains sentiment bearing words weighted within [-1; 1] interval, with -1 standing for negative and 1 for positive tonality. These words are adjectives, adverbs, verbs, and nouns. That is why these four POS and all their combinations will be considered as the features for Sentimal classifier. Table 3.1 shows all the combinations of features for Sentimal.

It will be analysed to which extend different POS of German language are indicative that the sentence is subjective and how the choice of particular parts of speech and their combinations influence the results of classification performance.

The central goal is to find out for Sentimal tested on each dataset the POS combination that produces the best classification results.

1	Adjectives
2	Adverbs
3	Verbs
4	Nouns
5	Adjectives + Adverbs
6	Adjectives + Verbs
7	Adjectives + Nouns
8	Adverbs + Verbs
9	Adverbs + Nouns
10	Verbs + Nouns
11	Adjectives + Adverbs + Verbs
12	Adjectives + Adverbs + Nouns
13	Adverbs + Verbs + Nouns
14	Adjective + Verbs + Nouns
15	Adjectives + Adverbs + Nouns + Verbs

Table 3.1: List of features to consider for Sentimal.

The following settings will be implemented as they are believed to be beneficial for classification task as well:

- Accounting for negations: The impact of negations in the sentences will be studied as well and it will be analysed to which extend implementation of the negation recognition method improve the classification performance.
- Accounting for emphasize: Another setting namely emphasize map (a list of words with corresponding emphasize weight) will be used as well as it might improve the classification performance. The list of emphasize words used in this master thesis is present in Appendix A.

The concrete impact of these feature combinations will be discussed in Chapter 4.

3.2.2 Preprocessing and classification algorithm

As standard preprocessing techniques, Sentimal applies POS tagging, stopwords removing, and words splitting (tokenizing) see Figure 3.5.

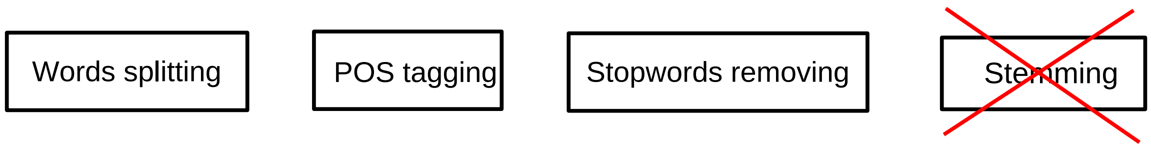


Figure 3.5: Sentimal's preprocessing elements.

Stemming is not used in the preprocessing pipeline. As was mentioned before Sentimal uses for classification task SentiWS which lists all sentiment bearing words together with their inflections if applicable. So all the sentiment words and word inflections can be directly looked up in the sentiment analysis dictionary. That is the reason why stemming is not considered.
The preprocessing and classification algorithm developed for Sentimal for every sentence consists of eight steps which are described in Figure 3.6.

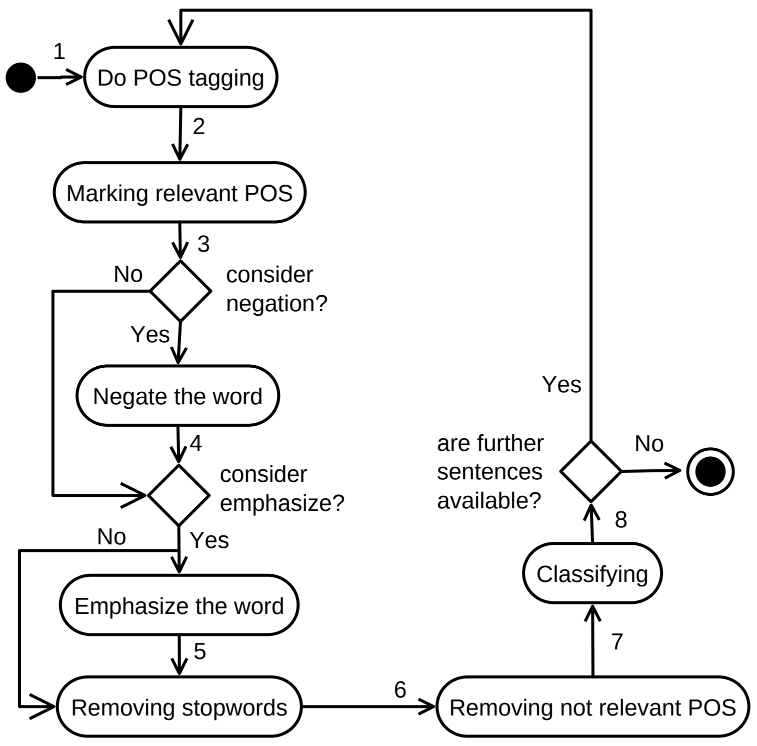


Figure 3.6: Activity diagram for Sentimal's preprocessing and classification algorithm.

All the steps in the algorithm should be considered in a given order for providing correct classification performance.

Let us apply the steps of the classification algorithm to a simple sentence:

„Nicht Junge, sondern ein schönes Mädchen spielt mit dem Auto“**.**

Condition. Let us consider only adjectives and account for negations, and emphasize.

1. First the POS tagging is applied to the sentence.

Nicht (PRTC) Junge (NN) sondern(CJ) ein (ART) schönes (ADJ) Mädchen (NN) spielt (VV) mit (PRP) dem (ART) Auto (NN)“.

Where PRTC means particle; NN –noun; CJ- conjunction, ART- article; ADJ-adjective, VV – verb, PRP – preposition.

2. In the second step we mark only adjectives, as for this task only adjectives are going to be considered.
   Nicht (PRTC) Junge (NN) sondern(CJ) ein (n) schönes **(ADJ)** Mädchen (NN) spielt (VV) mit (PRP) dem (m) Auto (NN)“.
3. We check if the word *schönes* is negated (due to the developed algorithm two words before the marked POS are taken into consideration). As we can see the word *schönes* is not negated in this example.
4. We check if the word *schönes* is emphasized (one word before the considered POS is taken into account). As a result the word *schönes* is not emphasized.

5. Now we can remove the words from the sentence that belong to the stopwords. As a result we will receive

Nicht (PRTC) Junge (NN) schönes **(ADJ)** Mädchen (NN) spielt (VV) Auto (NN)".

6. As the next step we remove all parts of speech except of those, marked in Step 2 (in this examples POS to consider are adjectives) and we obtain:

schönes (ADJ)

7. Then adjectives (in this example only one word) is going to be looked in the dictionary for its sentiment weight accounting for all preconditions – the word is not negated and the word is not emphasized.

As the result the classification algorithm would classify the sentence as being positive because the sentiment score of the sentence (the sum of weights of the all terms in the sentence) is positive.

Let us have a look what would happen if we would remove stopwords in Step 3 instead of Step 5.
In Step 3 after removing stopwords we would receive the following result
Nicht (PRTC) Junge (NN) schönes **(ADJ)** Mädchen (NN) spielt (VV) Auto (NN)".
Then we would check if the adjective *schönes* is negated – and in this case the adjective would be negated, which in the end would produce the wrong result i.e. the sentence would be classified as negative.
That is why on this small example it was shown the idea behind the classification algorithm developed for classifying sentences in Sentimal.

3.3 Palladian Text Classifier

In this section the principles of PTC's performance will be introduced. First of all, the considered set of features and additional settings which form experimental setup will be introduced in Subsection 3.3.1. Subsection 3.3.2 discusses in conceptual level the preprocessing techniques and classification algorithm used. In Subsection 3.3.3 the implementation details of PTC performance will be described. Figure 3.7 shows general structure of PTC.

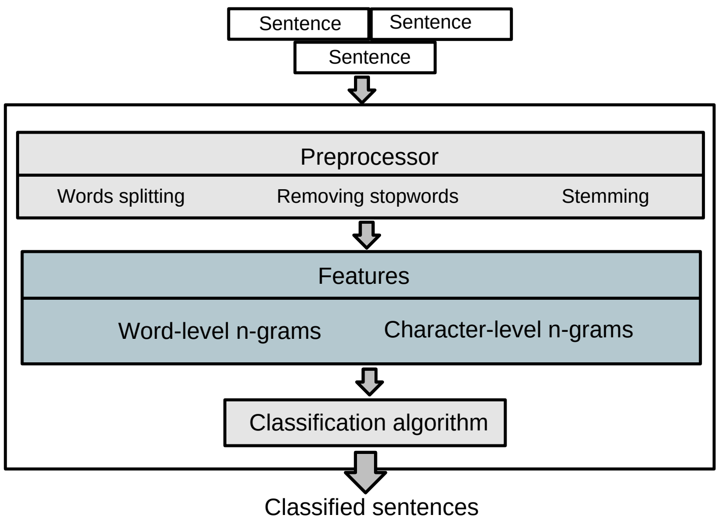


Figure 3.7: The structure of Palladian Text Classifier.

As can be seen from Figure 3.7, PTC receives as an input sentences out of which to be described in Section 3.3.1 features are derived. Before actual classification process takes place some preprocessing techniques like stemming and stopwords removing will be applied.

3.3.1 Experimental setup

In the following features that will be applied to PTC are described. The features which can be derived from the input sentences in PTC are n-grams, namely word-level n-grams and character-level n-grams. One of the master thesis goals is to define the feature set that produces the best classification performance. Due to the fact that it is impossible and not reasonable to test all possible variants of features combinations, in this master thesis word and character n-grams will be tested as long as incrementing the value of n-grams will still have some positive influence on classification results. The combinations of features considered for PTC are depicted on Table 3.2.

n-grams type	Value of n [min, max]

Word n-grams	1; [1,2]; [1,3]; [1,4]; [1,5]; [1,6]; [1,7]; [1,8]; [1,9]; [1,10];
	[2,3]; [2,4]; [2,5]; [2,6]; [2,7]; [2,8]; [2,9]; [2,10];
	[3,4]; [3,5]; [3,6]; [3,7]; [3,8]; [3,9]; [3,10];
	[4,5]; [4,6]; [4,7]; [4,8]; [4,9]; [4,10];
	[5,6]; [5,7]; [5,8]; [5,9]; [5,10]

Char n-grams	[1,2]; [1,3]; [1,4]; [1,5]; [1,6]; [1,7]; [1,8]; [1,9] ... [1;40]

Table 3.2: List of features to consider for Palladian Text Classifier.

The influence of these features on classification performance will be first tested when no preprocessing techniques are considered, then when stemming is applied, and afterwards when both stemming and stopwords removing are considered. Analysing all classification result will enable us to answer the question if these techniques really improve the classification performance of PTC, tested and trained on three datasets.

3.3.2 Preprocessing and classification process of PTC

Figure 3.8 shows the preprocessing techniques that are used in PTC.

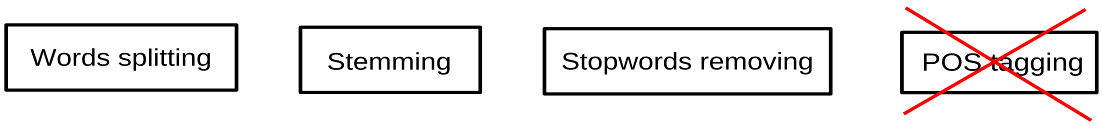


Figure 3.8: PTC's preprocessing elements.

POS tagging is not applied because PTC considers all POS when unigrams are considered as features. PTC builds for every word (n=1) or higher level n-grams at training stage a dictionary, by counting and normalizing the co-occurrences of a feature and a category.
PTC is based on supervised learning. It means that the classifier first should be training on the training dataset and afterwards the performance of it should be estimated using testing dataset. The preprocessing and classification algorithm developed for PTC consists of thirteen steps which are depicted in Figure 3.9.

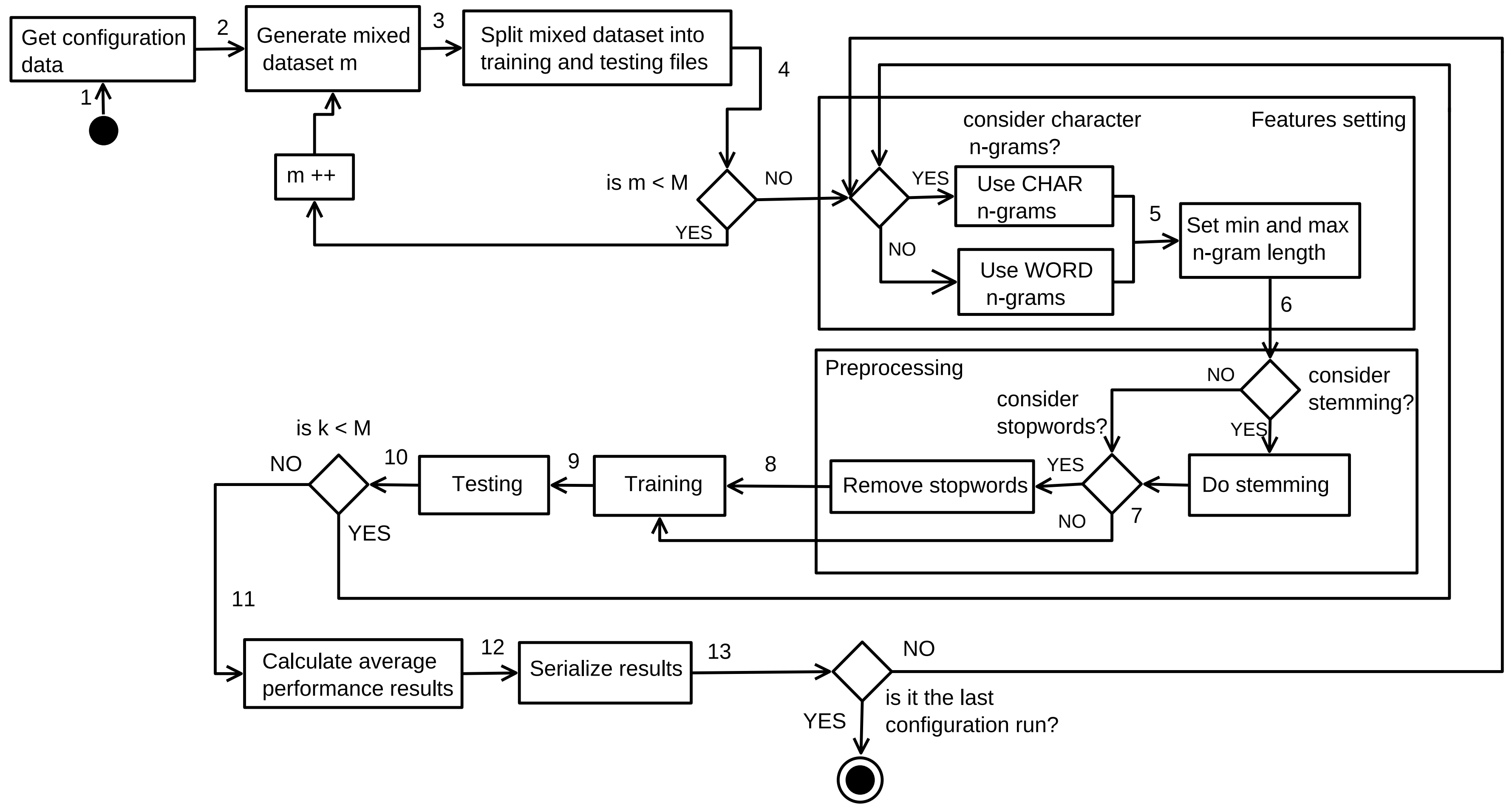


Figure 3.9: PTC’s preprocessing and classification algorithm.

In Step 1 the configuration file is deserialized, in this file all features are defined for the selected by researcher number of configuration runs. In Step 2 randomly mixed dataset is generated out of the input dataset, in Step 3 this mixed dataset is split into two files, namely training and testing. Steps 2 to 3 are repeating in a loop till the condition m < M is satisfied, where m is a parameter, and its value is incremented after each iteration and M is the maximum number of randomly mixed datasets derived from the input dataset. First m is set to 0 and the upper M is defined by researcher. Before breaking through the loop, we will obtain M training and M testing datasets, which will be needed for M times training and M times testing the classifier in a loop with the goal to calculate the average classifier performance for M runs which is done in Step 11 for current configuration run. This average performance helps to avoid fluctuations of performance results, as the classification results strongly depend on the training and testing data. Once M training and M testing datasets are generated, the methods for randomly mixing input dataset and splitting it into training and testing files can be commented in the code, while PTC for every new configuration run is trained and tested on the same M training and M testing datasets. It enables to provide fair comparison of the influence of the applied features on classification performance. Step 4 and Step 5 cares for the features selection, namely word or character-level n-grams and the MIN and MAX value used for these features, the concrete values of which are specified in the configuration file for each configuration run. Step 6 and Step 7 realize preprocessing pipeline. During Step 8 and Step 9 actual training and testing take place. And in Step 10 the decision if to loop through these algorithms starting with Step 4 again based on the value of parameter *k* is made. The looping takes place till the condition *k < M* is satisfied. Where *k* is a parameter and M is the number of runs (number of times the algorithm loops through the Steps 4 to 10) using results of which the average PTC performance is calculated for the current configuration run. Step 11 averages the classification results and in Step 12 the classification results of PTC for current configuration run are serialized e.g. to CSV file. In the final Step 13 the algorithm checks if one more configuration run is available. If it is the case, the algorithm loops starting from the Step 4 applying the settings of the following configuration run. The algorithm terminates if no further configuration runs are present.

The concrete execution path is defined by XML[4] configuration file, where all features and preprocessing elements can be switched on and off.

3.4 Implementation

In this section the implementation details of Sentimal and PTC classifiers are presented.

In a first step, the selected technologies, namely the programming language and development environment are presented in Subsection 3.4.1. Subsection 3.4.2 gives an introduction to the Palladian toolkit. Then the implementation details of Sentimal are presented in Subsection 3.4.3. Subsection 3.4.4 presents the implementation details of PTC.

3.4.1 Applied technologies

The object-oriented language Java was used for implementing of the Sentimal's and PTC's classification algorithms. This language was on one hand determined by the fact that the Palladian toolkit is written in Java and on the other hand by its generality and simplicity of use. As a

[4] http://www.w3.org/standards/xml

development environment the Eclipse IDE[5] was used. The build automation and software management tool Apache Maven[6] was used for building and managing the two software projects. The functionality of Palladian toolkit was used extensively for enabling Sentimal's and PTC's realization.

3.4.2 Palladian toolkit

The Palladian toolkit was created for realizing recurring Internet Information Retrieval tasks such as crawling, classification, and extraction of various types of information (Urbansky, Muthmann, Katz, 2011).

Palladian toolkit is utilized in this work as it includes a lot of algorithms for document classification tasks, that can be used and upon which new classification algorithms can be developed.

3.4.3 Sentimal implementation

In this section the implementation details of Sentimal classifier are presented. First the main packages, classes, and methods of Palladian used for enabling Sentimal's functionality are presented in Subsection 3.4.3.1. Then in Subsection 3.4.3.2 the new developed classes are described and the classification algorithm is visualized using Unified Modeling Language (UML)[7] sequence diagram.

3.4.3.1 Palladian's packages and classes employed

The list of the main packages and classes of Palladian toolkit which were utilized in this master thesis for implementing Sentimal are presented in the following.

The following groups of packages has been used:

- *ws.palladian.classification*[8]

The packages of this group include classes and methods for enabling classification of documents. Among the most important classes used for providing Sentimal's functionality are *GermanSentimentClassifier, CategoryEntry* and *Stopwords:*

  - *GermanSentimentClassifier* is a class which can be used for classifying sentences into positive and negative tonalities. In this master thesis one of its methods namely *classify* was extended and applied in Sentimal's classification algorithm.
  - *CategoryEntry is a class which holds information how relevant a certain category for a word is.*
  - *Stopwords* provides the logic for accessing stopwords lists.

- *ws.palladian.preprocessing*[9]

This group of packages realizes classes and methods that enable the preprocessing steps such as removing stopwords, stemming, POS tagging. It provides a lot of functionality helpful for natural language processing tasks.

---

[5] http://www.eclipse.org
[6] http://maven.apache.org
[7] http://www.omg.org/spec/UML/2.3/Infrastructure/PDF/
[8] Among the applied packages that belong to this group are: ws.palladian.classification, ws.palladian.classification.page, ws.palladian.classification.sentiment.
[9] To this group belong the packages: ws.palladian.preprocessing, ws.palladian.preprocessing.nlp, ws.palladian.preprocessing.nlp.pos

Among the main classes used from this package are:

  - *OpenNlpPosTagger* for splitting the sentences into words and tagging each word with its corresponding POS tag.
  - *TagAnnotation* class, that allows access to the tokens and tags of the annotations and
  - *StopWordRemover* class which removes a set of words which were specified as stopwords from the to be classified document.

- *ws.palladian.helper*[10]

To this group of packages belong the classes that provide additional functionality, such as algorithms that supply mathematical functionality as well as additional functionality for file and storage operations.

To the classes, used in this master thesis from this package belong the following: *ConfusionMatrix, FileHelper,* and *CollectionHelper.*

3.4.3.2 Classes created for Sentimal realization

The core of Sentimal classifier performance is formed by the developed classes *Sentimal*, *Preprocessor*, *ConfigurationDeserializer* and *PreprocessedEntity*.

The class *Sentimal* is the central and initial point of program logic. It was developed with the overall goal to classify input sentences into three tonalities, namely positive, negative, and neutral. It utilizes a model which was generated using SentiWS[11] dictionary. This model includes a map of sentiment words and their inflections, and the corresponding to each word sentiment score and is used in a final classification step.

ConfigurationDeserializer is a class developed for deserializing configuration data, specified by the researcher to determine the execution behavior. This behavior is provided via an XML file that determined the settings for all intended runs of Sentimal. For every run the applied features are specified within this XML configuration file. An example of the values provided for one run is presented in Figure 3.10 (Features: adjectives, adverb, nouns, and verbs; considering negation and emphasize).

```xml
<classificationrun>
    <termtype>ADJ</termtype>
    <termtype>ADV</termtype>
    <termtype>N</termtype>
    <termtype>V</termtype>
 <!-- valid values: true and false -->
       <considernegation>true</considernegation>
       <useemphasize>true</useemphasize>
</classificationrun>
```

Figure 3.10: Possible features and parameters which can be applied to Sentimal and set via XML configuration file.

After Sentimal receives the specified configuration data, it initializes the Preprocessor object, passing to it all features and parameters specified in the configuration file.

[10] To this group belong packages: ws.palladian.helper, ws.palladian.helper.collection, ws.palladian.helper.math.

[11] http://wortschatz.informatik.uni-leipzig.de/download/sentiws.html

The code is running in two loops, the first loop considers settings for all defined configuration runs, the second loop is for going through all the sentences in the dataset and classifying them.

The class Preprocessor realizes all preprocessing steps defined in Chapter 3.2.2 using the configuration options provided within the configuration file. It initializes the work of Tokenizer and PosTagger, thus splitting the input sentences into words and assigning to every word the corresponding POS tag. Afterwards, it marks all relevant POS and thus POS, mentioned in the configuration file for the currently applied run and checks if the words, corresponding to relevant POS are negated and emphasized. Afterwards, it removes all stopwords from the sentence. This execution order is important due to the fact that potentially defined Stopwords may have an influence on the decision if a word is negated or emphasized. As the final step, the method removeNotRelevantPOS() is executed which returns an ArrayList<PreprocessedEntity> containing only those words and their characteristics (negated, emphasized) which should be considered during the classification step based on their POS category.

Afterwords, Sentimal invokes the classify method, passing to it the ArrayList<PreprocessedEntity>. This extended classify method returns a CategoryEntry, which indicates to which class the sentence belongs. In a final step, the ConfusionMatrix is constructed by Sentimal and written to a CSV file. This CSV file is accessed for analyzing purposed by the author of this work using Microsoft Excel[12].

The core steps of the sentences' classification process via Sentimal are visualized in a simplified manner in Figure 3.11.

[12] http://office.microsoft.com/de-de/excel

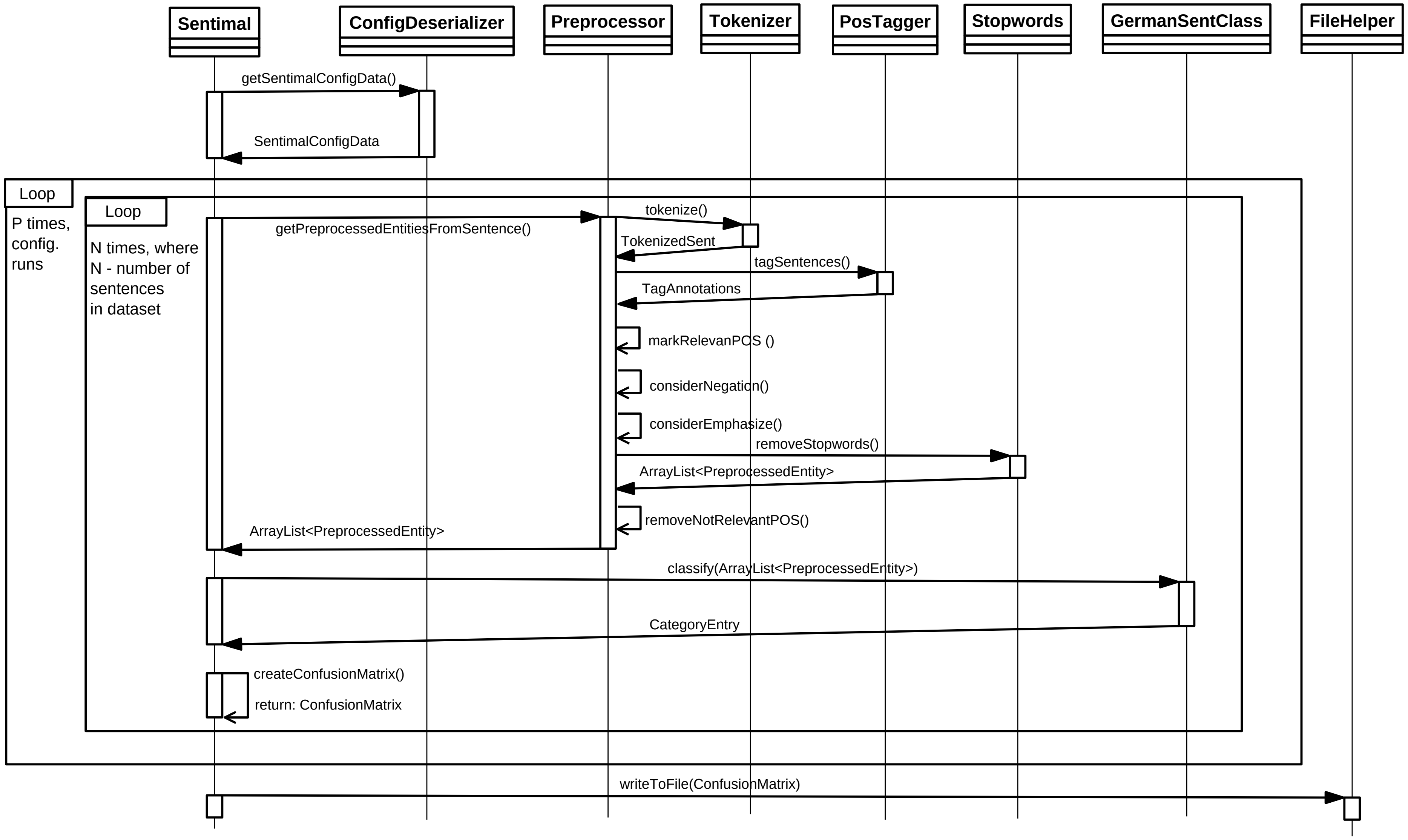


Figure 3.11: Sequence diagram of Sentimal's performance.

3.4.4 PTC implementation

In this section implementation details of realizing PTC are presented. In Subsection 3.4.4.1 the utilized in this work classes of Palladian toolkit are briefly described. Subsection 3.4.4.2 describes the developed classes and visualize implementation process via UML sequence diagram.

3.4.4.1 Classes used from Palladian toolkit

The functionality of Palladian toolkit (Urbansky, Muthmann, Katz, 2011) was utilized for providing the functionality of PTC as well.

Among the most important groups of packages utilized are:

- *ws.palladian.classification*[13], from which the following classes like *DictionaryClasifier, ClassifierEvaluator,ClassificationTypeSettings, FeatureSetting, Dataset, ClassifierPerformanceResult* were utilized for building PTC functionality
- *ws.palladian.helper*[14], from which *FileHelper, DatasetManager, ConfusionMatrix,* and *MathHelper* classes were applied
- *ws.palladian.preprocessing*, from which such classes as *StopWordsRemover* and *ProcessingPipeline* we utilized.

3.4.4.2 Developed classes for PTC

The core classes developed for implementing PTC are *PalladianTextClassifierEvaluator* (PTC), *ConfigurationDeserializer,* and *ConfigurationData.*

ConfigurationDeserializer is a class developed for deserializing the data from XML configuration file. Figure 3.12 shows an example for the features and parameters which can be set for PTC via the XML configuration file (Features: word or character n-grams; considering stemming and considering stopwords removing; min and max n-gram length; percentage of data used for training where 0.7 means 70% of data used for training).

```
<classificationrun>
<!-- valid values: true and false -->
    <usingstemmer>true</usingstemmer>
    <usingstopwords>true</usingstopwords>
    <usingcharngrams>false</usingcharngrams>
       <usingwordngrams>true</usingwordngrams>
      <minvalueforngram>1</minvalueforngram>
      <maxvalueforngram>2</maxvalueforngram>
<percentagetraining>0.7</percentagetraining>
</classificationrun>
```

Figure 3.12: Features and parameters which can be applied to PTC and set via XML configuration file.

[13] The applied packages that belong to this group are: ws.palladian.classification, ws.palladian.classification.page, ws.palladian.classification.page.evaluation.
[14] The packages used from this group are: ws.palladian.helper, ws.palladian.helper.math.

In order to provide this functionality, M datasets were randomly generated from the input dataset with the help of *MathHelper* class in a first step. After this, these M randomly mixed datasets are written to a CSV file and afterwards each of them was split by *DatasetManager* class into two files, namely Trainingdataset.csv and Testingdataset.csv, which were applied for training and testing the classifier. The percentage of data used for creating Trainingdataset.csv. is determined by researcher via configuration file. This process is depicted in the sequence diagram in Figure 3.13.

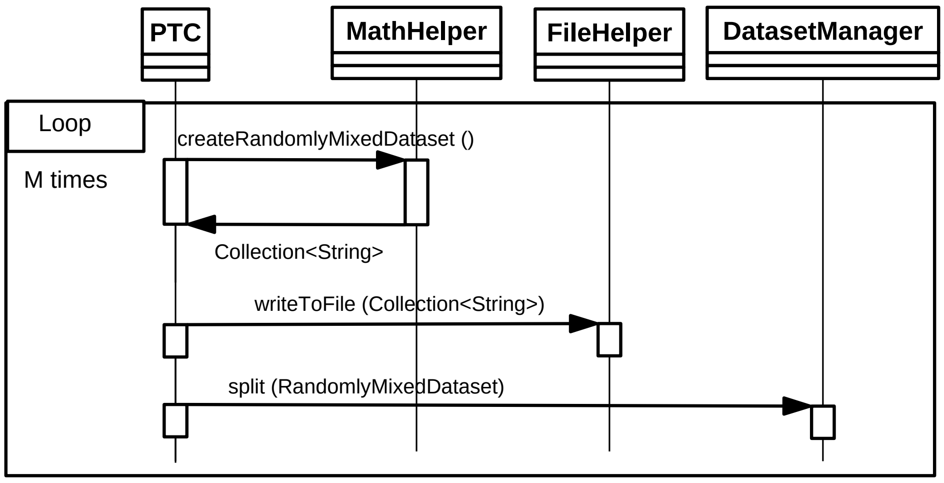

PTC
MathHelper
FileHelper
DatasetManager
Loop
M times
createRandomlyMixedDataset ()
Collection<String>
writeToFile (Collection<String>)
split (RandomlyMixedDataset)

Figure 3.13: Creating randomly mixed dataset and splitting it into training and testing datasets.

In this thesis 10 datasets were randomly generated which were split into 10 training and 10 testing datasets using the process described above. It was necessary for valid classification results to randomly generate mixed datasets and to calculate average classification results for all 10 runs, due to the fact that PTC performance depend on the training and testing data used. It helped to avoid fluctuations in the results.

The *PTC* class works as the following. After deriving configuration data, PTC invokes method of *MathHelper* class for generating randomly mixed datasets (see Figure 3.13). Afterwards *PTC* invokes methods of *FeatureSetting* and *ProcessingPipeline* classes in order to set the features, specified by researchers in the configuration file for the current configuration run. After features and preprocessing mechanisms are set, *PTC* invokes the training method of *DicionaryClassifier* class and trains the Dictionary classifier on the generated training dataset. This method returns TrainedDictionaryClassifier. As the next step, *PTC* applies testing (evaluating) of the trained classifier on the generated test dataset. After looping through this process 10 times we can derive a list of performances results List<ClassifierPerformanceResult> which is passed to *ClassifierEvaluator* class, which calculates the average performance and returns the result ClassifierPerformanceResult. This result is written to a CSV file by invoking writeToFile() method of *FileHelper* class. Afterwards the procedure repeats for the next configuration run. Created CSV files are accessed for analyzing purposed by the author of this work using Microsoft Excel. The discussed process is visualized by the sequence diagram in Figure 3.14.

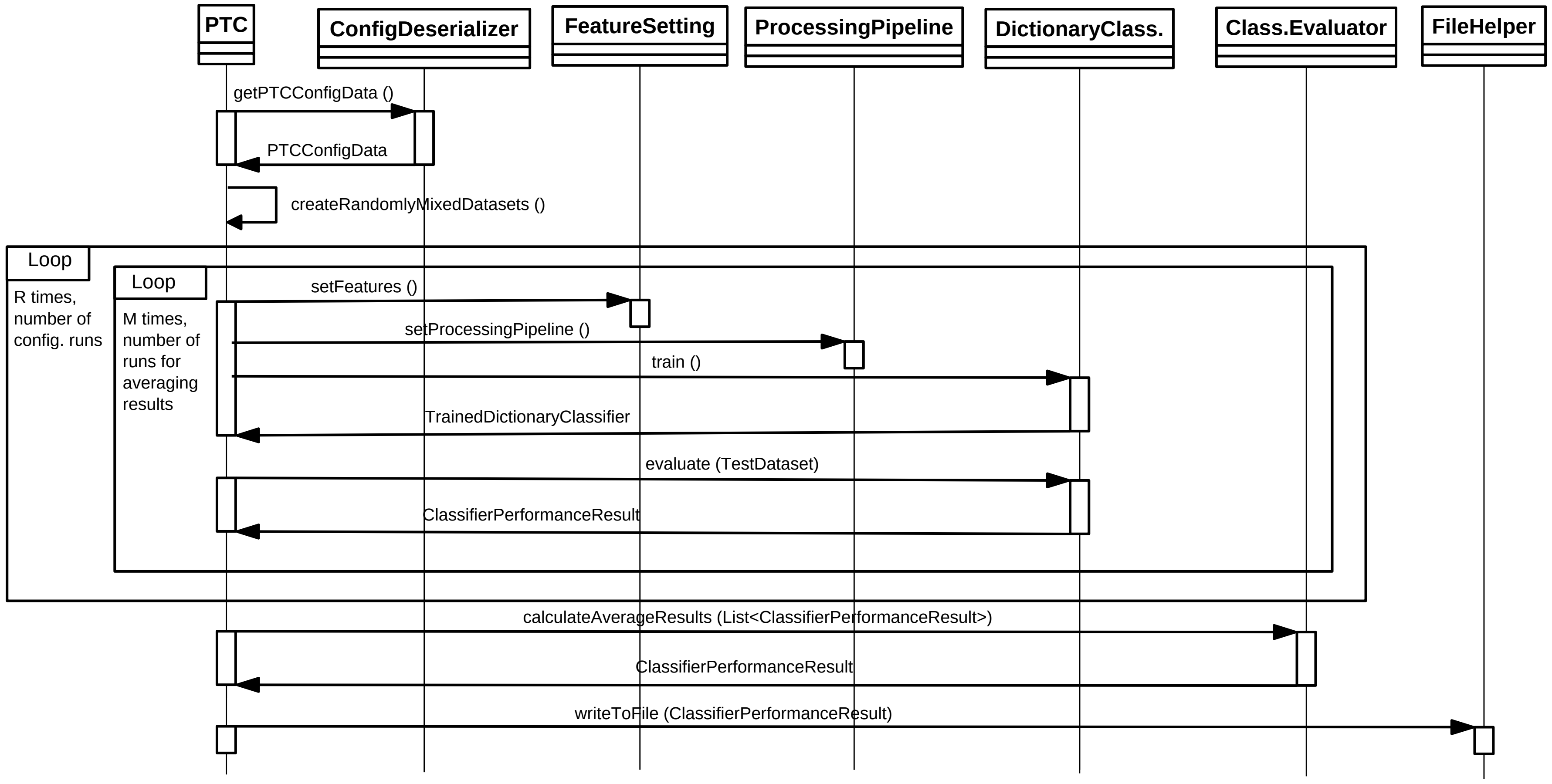

PTC
ConfigDeserializer
FeatureSetting
ProcessingPipeline
DictionaryClass.
Class.Evaluator
FileHelper
getPTCConfigData ()
PTCConfigData
createRandomlyMixedDatasets ()
Loop
R times, number of config. runs
Loop
M times, number of runs for averaging results
setFeatures ()
setProcessingPipeline ()
train ()
TrainedDictionaryClassifier
evaluate (TestDataset)
ClassifierPerformanceResult
calculateAverageResults (List<ClassifierPerformanceResult>)
ClassifierPerformanceResult
writeToFile (ClassifierPerformanceResult)

Figure 3.14: Sequence diagram of PTC performance.

3.5 Summary

In Section 3.1 the general overview of experimental workflow was presented.

Section 3.2 introduced us to the experimental setup of Sentimal, and presented developed for Sentimal preprocessing and classification algorithm which was visualized by an activity diagram.

Section 3.3 presents the experimental setup of PTC, describes PTC learning process which is visualized by activity diagram.

In Section 3.4 the implementation details of realizing Sentimal and PTC are presented and performance of both classifiers is described applying sequence diagrams.

Chapter 4: Evaluation

This chapter evaluates the results of the realized analysis and addresses the research questions stated in Chapter 1.

In Section 4.1 a detailed description of the applied datasets is presented. Section 4.2 defines optimal dataset characteristics for learning PTC. This forms the foundation for the remainder of this chapter and should enable the reader to understand the characteristics of the datasets and thus the research results in detail. In a next step, the research questions forming the basis for this work will be addressed. Thus, the following three aspects will be focused:

1. The results of PTC and Sentimal classification performances for all considered features will be discussed and evaluated, and the best configuration for each classifier identified in Section 4.3. It will be also discussed, whether, firstly, the classification results of PTC and Sentimal are skewed towards any particular opinion tonality and, secondly, whether they are domain independent.
2. The comparison of the classification results of both classifiers will be conducted in Section 4.4.
3. After evaluating the performance results of both classifiers, the content analysis of three datasets will be realized in Section 4.5.

The chapter concludes by giving a short overview of the central evaluation results.

4.1 Dataset description

Three datasets were established and have been used for evaluating the classification performance of both classifiers. These datasets have been constructed from completely different Web resources covering different domains. The performance evaluation using these respective datasets will enable to discover whether the tonality-based classification is domain specific or not. The characteristics of the three datasets will be discussed in the following sections.

4.1.1 First dataset (Net-clipping)

The first dataset has been constructed from the articles taken from the net-clipping database. Net-clipping[15] is a Web and Social media monitoring tool keeping track of what is said about companies' products all over the Web. It monitors 6.500 German language online-media and different social media resources such as Blogs (45.000), forums in the Web (8.000), twitter, Facebook, and YouTube. It gives customers the possibility to know who is writing about their products online and which opinions are expressed, and it also keeps track of the most recent comments concerning the products.

The articles for establishing this dataset were collected within a three days interval from a variety of Internet resources such as Web blogs, Facebook, Wikipedia, news portals, and chat portals. All the topics which were discussed during those days were enclosed into these articles. Respective articles

[15] http://www.net-clipping.de/

were manually processed by the author of this thesis and split into sentences which were classified according to the opinion they carry i.e. positive, negative, or neutral. In a next step the manually classified (tagged) sentences were saved in a local repository. This repository serves as an input for the classifier evaluation.
The pie chart of sentences' distribution for the Net-clipping dataset without skewing them for having exactly the same number of positive, negative, and neutral sentences is depicted in Figure 4.1.
During the manual classification of the articles it was revealed that a majority of sentences are neutral and thus they do not carry opinions on a subject. It means that the data from the source systems include as a major part objective and as minor opinionated content. The overall sum of sentences in the first dataset is more than 1.600.

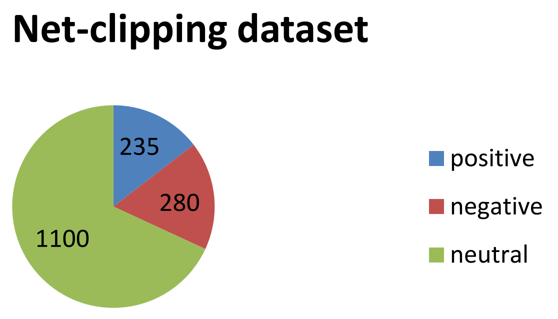


Figure 4.1: Sentence tonality distribution in Net-clipping dataset.

4.1.2 Second and third datasets

The second and third datasets were established by Michéle Sprejz (Sprejz, 2012) based on Amazon product reviews. These two datasets are separately saved in a local repository. The articles from Amazon include mainly opinionated content. Due to this the sentence distribution is completely different from the distribution in Net-clipping. The product reviews cover two domains: Mobile phones and Notebooks. These articles were manually split into sentences and classified into the same three categories as the Net-clipping dataset (positive, negative, neutral).

The sentences' tonality distribution for the domain Mobile phone is shown in Figure 4.2 and for the Notebook domain in Figure 4.3. These pie charts reveal the sentence distribution in the way they were collected from the Amazon product reviews without skewing them for having the same number of positive, negative, and neutral sentences. From Figure 4.2 and Figure 4.3 we can infer that the number of positive sentences outweighs the number of negative and neutral sentences.

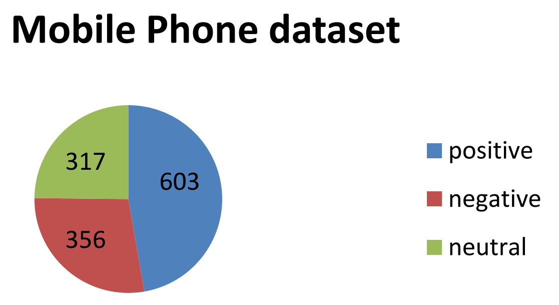


Figure 4.2: Sentence tonality distribution in Mobile phone dataset.

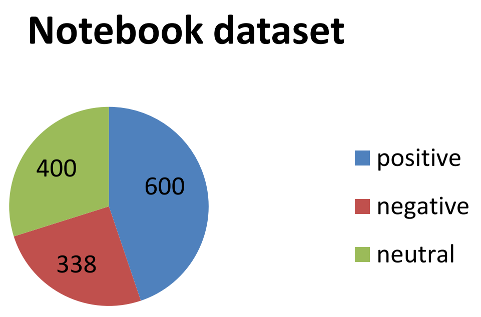


Figure 4.3: Sentence tonality distribution in Notebook dataset.

4.2 Optimal dataset characteristics for learning PTC

In order to provide a fair comparison of the two classification approaches it is essential to determine the optimal learning approach for the Palladian Text Classifier. For this, it has been analysed, which characteristics the dataset used for learning the classifier should have in order to achieve optimal classification results. In detail, the following questions should be answered:

- How should the tonality of sentences be distributed in the datasets?
- How many sentences should each dataset at least consist of?
- Which percentage of data should be used for training and testing the Palladian Text classifier?

These questions will be addressed in the following sections.

4.2.1 Tonality distribution in the datasets

In order to motivate the importance of the question for the optimal tonality distribution, preliminary classification runs will be discussed in a first step. This preliminary analysis has been carried out using one complete dataset for learning without applying stratification. The results of one of these analyses are displayed in Figure 4.4 (used dataset: Mobile phone). In the experiment presented in the following, 2 fold cross validation has been applied.

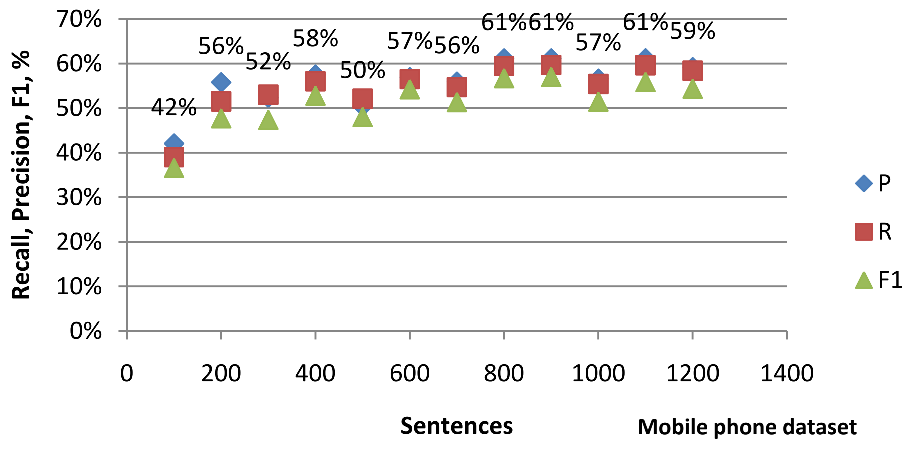
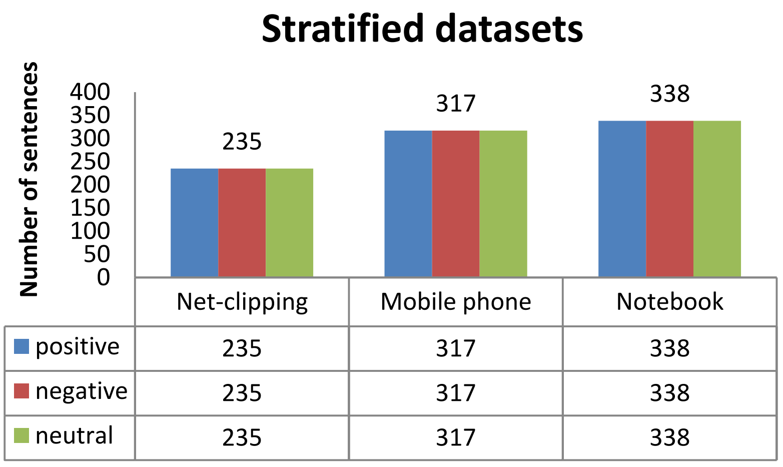


Figure 4.4: Classification behaviour of PTC tested on not stratified Mobile phone dataset.

The graph's X-axes in Figure 4.4 represents the number of input sentences and the Y-axes shows the value of precision, recall and F1-measure in percentage. The values of precision are marked in the figure. The performance results are calculated for every dataset split needed for cross validation, and whereupon they are summed and averaged, which provides the rationale for the F1 value lower than precision and recall, which we can observe in Figure 4.4. We can also see a rapid growth of precision when up to 200 sentences are used for training and testing. When 100 additional sentences are used, a decrement of performance can be seen. With up to 400 sentences precision improves again and then with 100 more sentences a decrement of precision of eight percent can be observed. This instable behaviour can be explained based on the characteristics of the applied dataset. As it was discussed above, the Mobile phone dataset includes 603 positive, 356 negative, and 316 neutral sentences. The discussed behaviour of fluctuating performance is an indicator that the classification results of PTC are especially fluctuating in the case of the dominant sentences of positive tonality. In order to avoid a too heavy impact of sentences of a single tonality class, the same amount of sentences from the three classes should be used. Thus, the datasets should be stratified.

Thus, for enabling a fair analysis, all three datasets were stratified. The resulting sentence distributions for the three employed datasets are depicted in Figure 4.5.



	Net-clipping	Mobile phone	Notebook
🟦 positive	235	317	338
🟥 negative	235	317	338
🟩 neutral	235	317	338

Figure 4.5: Sentence tonality distribution in Net-clipping, Mobile phone, and Notebook dataset.

Net-clipping dataset consists of 705 sentences (235 sentences of each tonality), Mobile phone includes 951 (317 sentences of each tonality), and Notebook 1014 sentences (338 sentences of each tonality).

4.2.2 Cardinality of datasets

In a second step it was analysed, how huge the datasets applied for learning the PTC classifier should be at least. In order to answer this question a set of experiments were realized. The idea of these experiments has been to learn the PTC incrementally increasing the number of sentences applied for the classifier. The classification performance in metrics of precision, recall, and F1 is expected to improve when the number of sentences is increased step by step, as the classifier learns new information from more training data. It is thus assumed that at some specific point the created model is saturated and stops learning additional information from training sentences and, consequently, classification results stay nearly constant. In this vein, the point at which further sentences used for learning do not have any impact on the classification results can be detected.

The results of experiments applied in order to show how many sentences in each dataset are needed for stabilizing classification behaviour of PTC are shown in Figures 4.6 to 4.8. The number of submitted sentences increases by 50 in every run. Figure 4.6 depicts the classification results for Net-clipping dataset. In the following presented experiment, 10 fold cross validation has been applied.

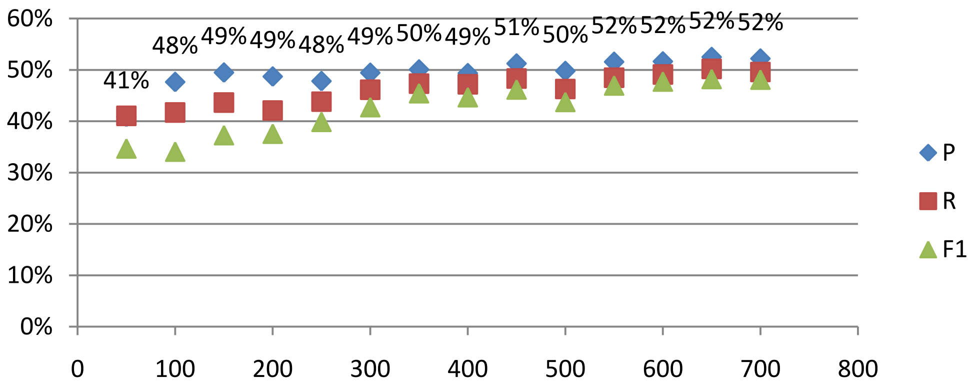


Figure 4.6: Checking how many sentences are needed for Net-clipping dataset.

The X-axis in Figure 4.6 represents the number of input sentences and the Y-axis shows the precision, recall and F1 in percentage. The values of precision are given in the figure. For testing purpose, the dataset of Net-clipping consisted of 705 sentences (235 sentences of each tonality). The number of input sentences for the classification process was initially set to 50 and then the classification algorithm was run 14 times in a loop, and, at each iteration, increasing the number of to be classified sentences by 50. In Figure 4.6 we can see that PTC for Net-clipping dataset requires at least 600 sentences. After 550 sentences the results of precision, recall, and F1 stabilize and by adding more sentences we almost do not achieve any increment in performance.

Figure 4.7 depicts the PTC classification behaviour if tested using the Mobile phone dataset. The values of precision are marked. Based on this test, we can derive the minimum amount of sentences needed for the learning process.

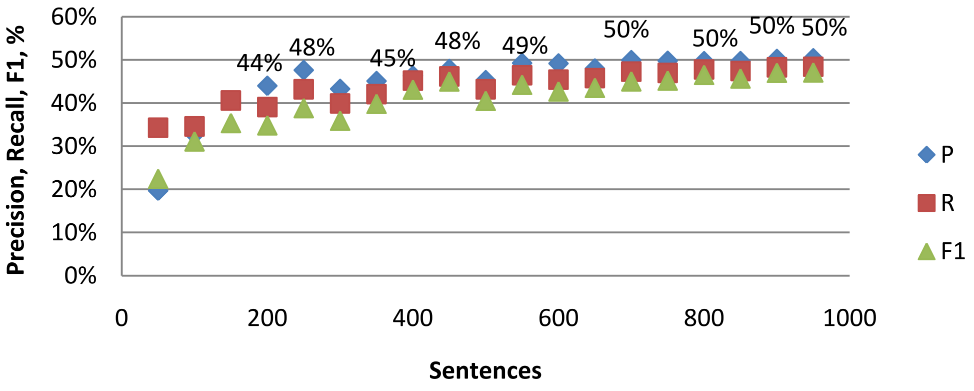


Figure 4.7: Classification results of PTC using the Mobile Phone dataset.

The stratified Mobile Phone dataset consists of 951 sentences. From the performance graphs we can infer that the performance stabilizes at about 900 sentences and nearly almost no increase in classification performance is achieved by adding further sentences.

In a last step, the classification behaviour of PTC if tested on Notebook dataset has been measured. The results of this test are depicted in Figure 4.8. The average performance results are calculated as well resulting in values of F1 lower than precision and recall. The values of precision are marked in the figure.

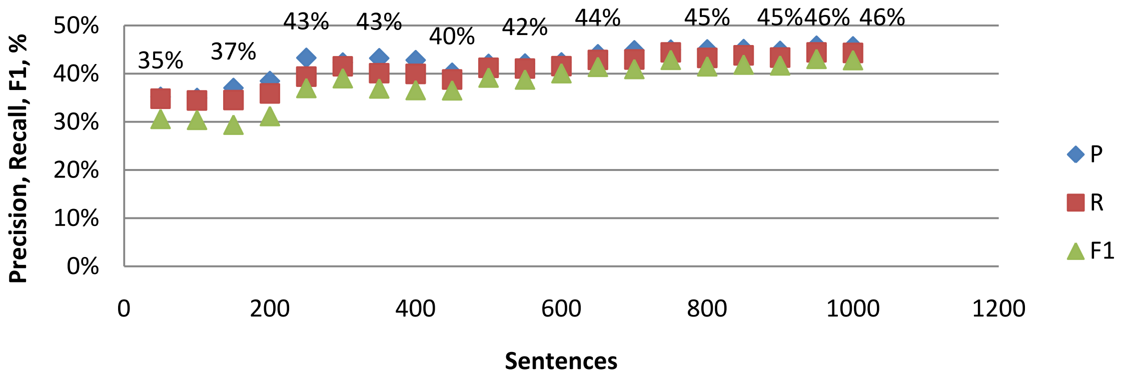


Figure 4.8: Classification results of PTC using the Notebook dataset.

The stratified Notebook dataset consists of 1,040 sentences. Using this set, we can observe a performance increment of 1 percentage after adding additional 50 sentences to a 900 sentences set, but after adding 50 further sentences, the precision and recall change only by a percentage of 0,1. Thus, we can assume that the classifier behaviour stabilizes at about 950 sentences.

From the applied classification performance tests we can conclude that for the Net-clipping stratified dataset 705 sentences, Mobile phone dataset 951 sentences, and Notebook dataset 1040 sentences are sufficient in order to carry out further analysis. Thus, the established three datasets can serve as a basis for the classification tasks.

4.2.3 Determining optimal percentage of training data and testing data for PTC

After the minimal cardinality of the datasets has been analysed, it is important to define the optimal percentage of sentences needed for training and testing PTC. For this purpose, PTC was configured to applying different settings of cross validation for learning.

1. 10-fold cross validation: With 10-fold cross validation only 10% of data is used for training and 90% for testing the trained classifier.
2. 4-fold cross validation: In this configuration, 25% of the data is used for training and 75% for testing the classifier.
3. 2-fold cross validation: In this setting, 50% of data is used for training and another 50% for testing the classifier, thus the dataset is split in two 2 files. The cross validation is done as follows: In a first step, the first of two files is used for training and the second for testing, and then vice versa, and the average results are calculated.

The performance results of evaluating PTC performance on three stratified datasets applying the three different settings (10-fold cross validation: cv=10, 4-fold cross validation: cv=4, 2-fold cross validation: cv=2) using word level n-grams ($1 \leq n \leq 2$) can be seen in Figures 4.9-4.11.

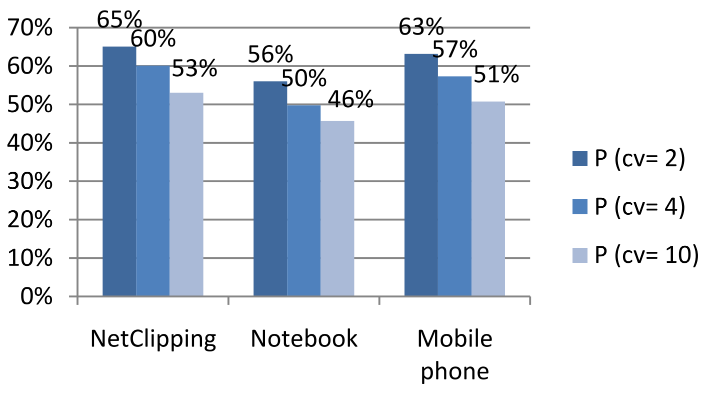


Figure 4.9: Precision of learning PTC on three datasets with different cross validation parameters.

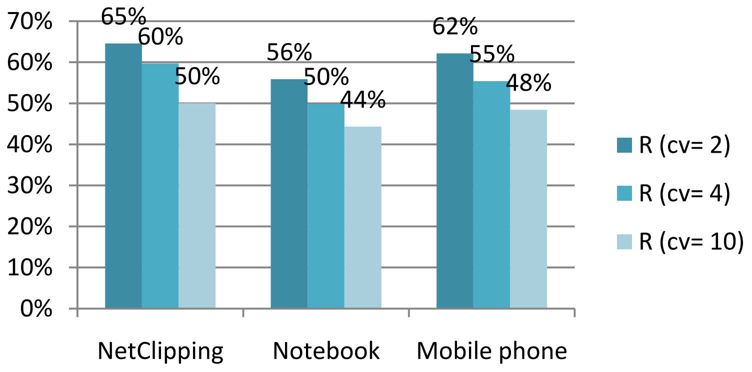


Figure 4.10: Recall of learning PTC on three datasets with different cross validation parameters.

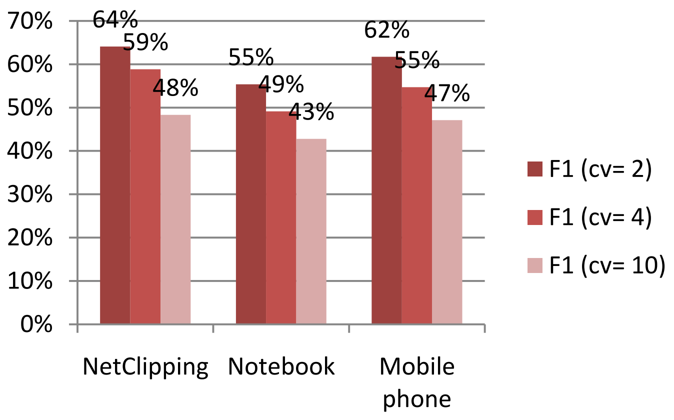


Figure 4.11: F1 of learning PTC on three datasets with different cross validation parameters.

It can be seen that the values of precision, recall and F1 improve significantly for all datasets if the amount of training data is increased from a percentage of ten (10-fold cross validation) to 50 (2-fold cross validation). For example, if the Mobile phone dataset is used for learning PTC applying 10-fold cross validation, the value of F1 is 47%, for 4-fold cross validation it is 54%, and for 2-fold cross validation it is 62%. An analogous behaviour can be observed in the case of the Net-clipping and Notebook datasets.

Due to the fact that the classifier performance improves if the percentage of training data is increased, it is important to observe how the classification results will change when incrementing the percentage of training data beyond 50%. As a result, another set of experiments has been realized. PTC was learned applying sentences from Net-clipping, Mobile phone, and Notebook datasets, varying the percentage of training data. At first, only 10% of data is used for training, then 20%, and so on till ultimately 90%. The central results of the experiment for each dataset can be seen in Figures 4.12 - 4.14 (Chosen features: unigrams).

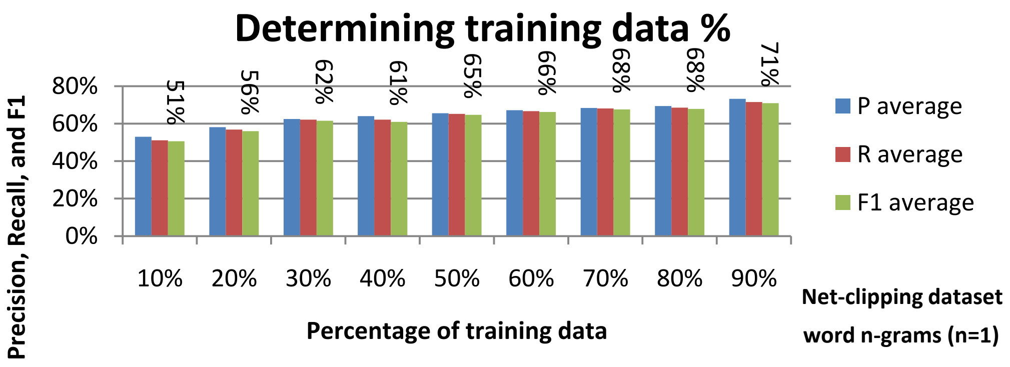


Figure 4.12: PTC performance based on varying percentage of training data (Net-clipping dataset).

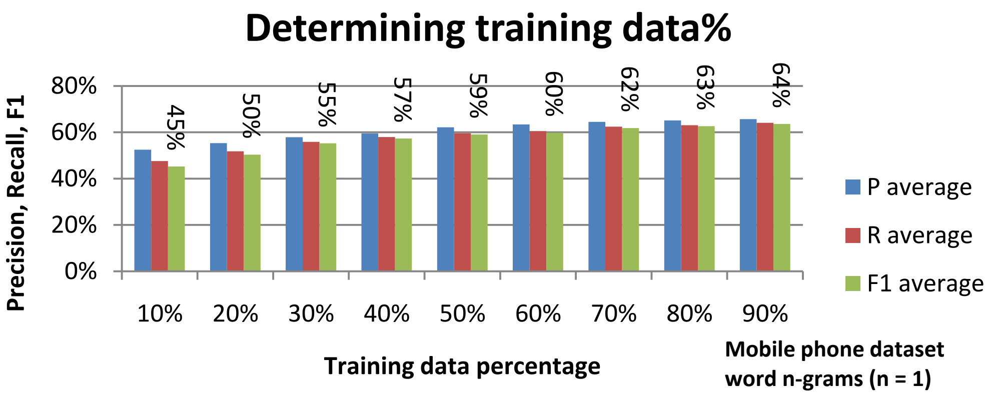


Figure 4.13: PTC performance based on varying percentage of training data (Mobile phone dataset).

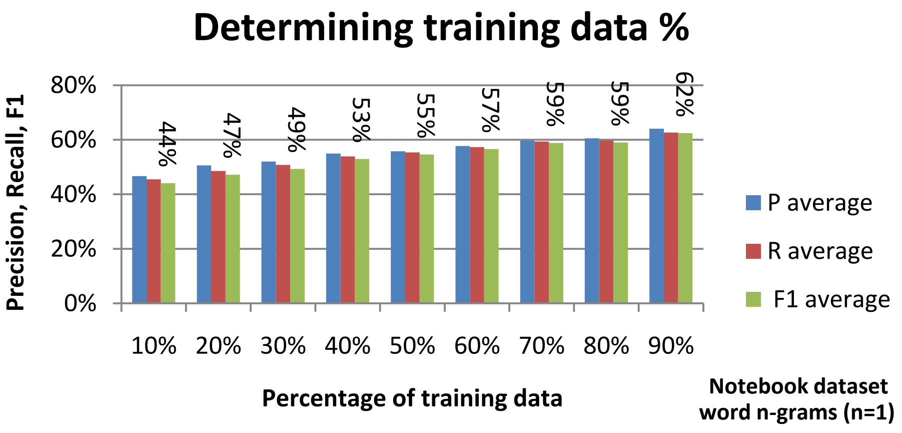


Figure 4.14: PTC performance based on varying percentage of training data (Notebook dataset).

In these figures the average value of F1 is marked with its percentage score. Analysing these figures, we can argue that PTC produces the best classification results when using 90% of sentences for training and 10% for testing, but in fact the classification performance for all three datasets is not much worse with 70% or 80% of training data.
If 90% of sentences from each dataset would be used for training, it is important to know whether the remaining 10% for testing are sufficient to reflect the classification behaviour of the full Net-clipping, Mobile phone, and Notebook datasets. We should have a look at some statistical measures such as sample size, confidence level, confidence interval, population size (Creative Research Systems, 2011) and calculate the amount of sentences required for testing in order to reflect the classification behaviour of the complete considered datasets.
In our case the **sample size** is the number of sentences needed to be left for testing in order to reflect the classification behaviour of all sentences in the given dataset.
The **confidence interval** (also called margin of error) is the plus-or-minus figure usually reported in newspaper or television opinion poll results (Creative Research Systems, 2011). For example, if a confidence interval is set to five and 33% of sample is classified as positive, it would mean that according to statistics between 28% (33-5) and (33+5) 38% of all sentences in complete dataset would be classified as positive.
The **confidence level** means how sure/certain we can be that the results will be located within the confidence interval. Abstracting to the sentence level and going back to the example above it would mean that when having the confidence level of 95% we could be 95% sure that any positive sentence classified by the classifier would be located within the confidence interval.
If the confidence level and the confidence interval are put together, it would mean that the person could be 95% sure that between 28%-38% of the sentences in the whole dataset would be classified as positive.

As most, researchers use the confidence level of 95% (Creative Research Systems, 2011). In this thesis the confidence level of 95% was chosen as well, and the confidence interval was set to 5%. After this, the sample size (number of sentences) needed to be left for testing was calculated. It was done for each of the three datasets accordingly.

The web-application (Creative Research Systems, 2011) for calculating the sample size of sentences when given the confidence level, confidence interval and the number of sentences in the datasets was used. The calculation results are depicted in Table 4.1.

Dataset Name	Number of sentences	Confidence level	Confidence interval	Sample size
Net-clipping	705	95%	5%	249
Mobile phone	948	95%	5%	274
Notebook	1014	95%	5%	279

Table 4.1: Calculation results for determining the sample size of the sentences for three datasets.

Three parameters determine the size of the confidence interval (Creative Research Systems, 2011): sample size, percentage and the dataset size. Thus, the larger the sample size is (number of sentences), the better it reflects the behaviour of a full dataset. So for a given confidence level, the larger the sample size, the smaller will be the confidence interval.
In the analysis, only stratified datasets are used and thus all three opinion tonalities are equally present in the datasets. It means that positive, negative, and neutral sentences are uniformly

distributed and we could argue that the next sentence classified will be a positive, negative, or neutral sentence in the worst case with the probability of 1/3. That is why the percentage parameter was set to 33% and the sample size was refined. Furthermore, the percentage of sentences needed for testing was calculated solving a simple proportion knowing the sample size and the number of sentences in the datasets. The results are presented on Table 4.2.

Dataset	Number of sentences	Confidence level	Confidence interval	Percentage	Sample size	Percentage for testing
Net-clipping	705	95%	5%	33 %	230	32,6% ~ 30%
Mobile phone	948	95%	5%	33 %	251	26,5% ~ 30%
Notebook	1014	95%	5%	33%	256	25,2% ~ 30%

Table 4.2: Calculation of the sample size and percentage of sentences needed for testing the classifier.

According to these calculations, we could conclude that at least 30% of the sentences in each of the three datasets have to be used for testing and the remaining 70% for training the classifier in order to satisfy the 95% confidence level and 5% confidence interval thus producing the classification results that reflect the classification behaviour of a full dataset.

Looking at Figures 4.12 – 4.14 we can see that PTC classification results are quite good when having 70% training and 30% testing data and almost the same (for some features slightly better or slightly worse) in comparison to the results if 80% of data are used for training.
Due to these results, the percentage of training and testing data will be fixed to 70% and 30% respectively in the analysis discussed in the following sections.

4.3 Features evaluation

In Chapter 3 a set of features for Sentimal and PTC was discussed. In this section we are going to estimate the performance of each of the classifiers on the considered feature set, define the best configuration for each of the classifiers and check if they are domain independent.

4.3.1 Sentimal features evaluation

In a first step, the features analysis, namely POS analysis will be realized. Secondly, the influence of considering negation and emphasizing on Sentimal's classification results will be tested and the best classifier configuration identified. Thirdly, classification results of Sentimal of three sentence tonalities will be evaluated. Fourthly, it will be analyzed how the peculiarities of used datasets influence Sentimal`s classification results, and fifthly, it will be checked if Sentimal is a domain independent classifier.

4.3.1.1 POS analysis

For Sentimal 45 configuration runs were tested. The first 15 configuration runs were realized in order to define how each part of speech in German language influences the classification performance. In every configuration run each POS, namely adjectives, nouns, verbs, adverbs, and all their combinations were considered.
Sentimal's classification results in metrics of precision, recall, and F1 evaluated using the Net-clipping dataset for all 15 POS combinations are depicted in Figure 4.15.

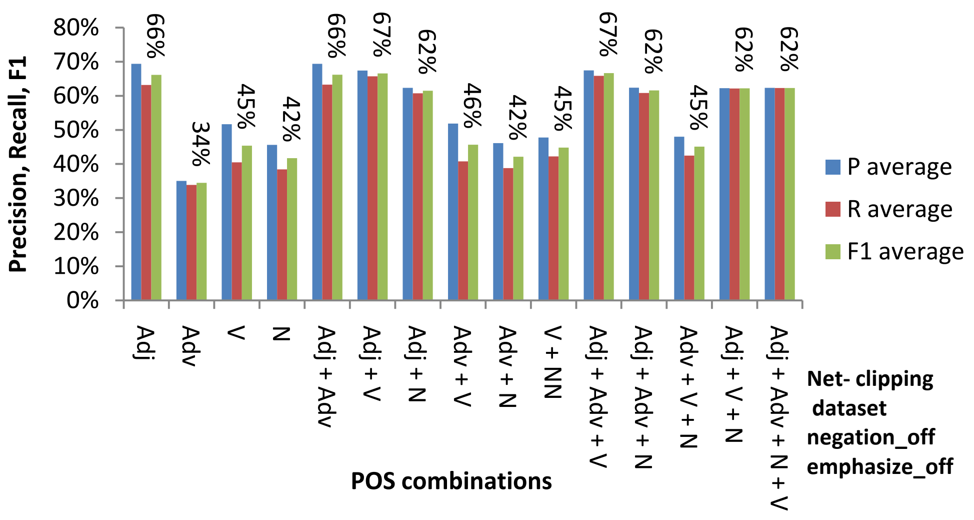


Figure 4.15: POS analysis using the Net-clipping dataset.

In Figure 4.15 the X-axis represents all POS combinations, where “Adj” stands for adjectives, “Adv” for adverbs, “V” for verbs and “N” for nouns. In the figure the average value of F1 is marked with its percentage score.

It can be seen in Figure 4.15 that adjectives, adjectives plus adverbs, adjectives plus verbs, and adjectives plus adverbs plus verbs appear to be the four best classification results in the case of the Net-clipping dataset. In each combination that shows good classification results, adjectives are present. This implies that adjectives (and adjectives in combination with other POS) are the best indicators of a sentiment in sentences.

Adverbs, verbs, and nouns alone show poor classification performance for the Net-clipping dataset. But considering adjectives together with other POS improve classification performance by 1%, though combination of all POS shows 5% worse performance in metrics of average F1, than for example the combination of adjectives plus verbs.

In order to check if the results presented so far can be generalized, the tests have been realized using the two further datasets. The classification results of first 15 experiments of Sentimal for the Mobile phone dataset is depicted in Figure 4.16.

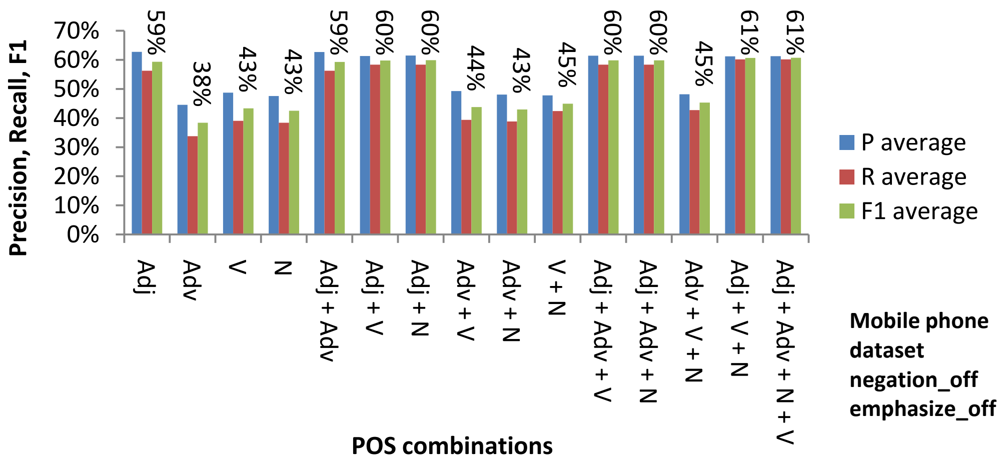


Figure 4.16: POS analysis using the Mobile phone dataset.

The results shown in Figure 4.16 proof the above mentioned statement that adjectives and their combinations with all POS are the best indicators of sentiment in sentences. In contrast of using the Net-clipping dataset, in the case of the Mobile Phone dataset Sentimal shows the best performance if all POS are considered. The value of average F1 score is 1% higher than if only adjectives plus verbs are taken into account.
Adverbs, verbs, and nouns alone show poor classification performance, but in combination with adjectives the classification results in metrics of F1 increase by two percent.

Figure 4.17 reflects the classification results of Sentimal in the case of the Notebook dataset.

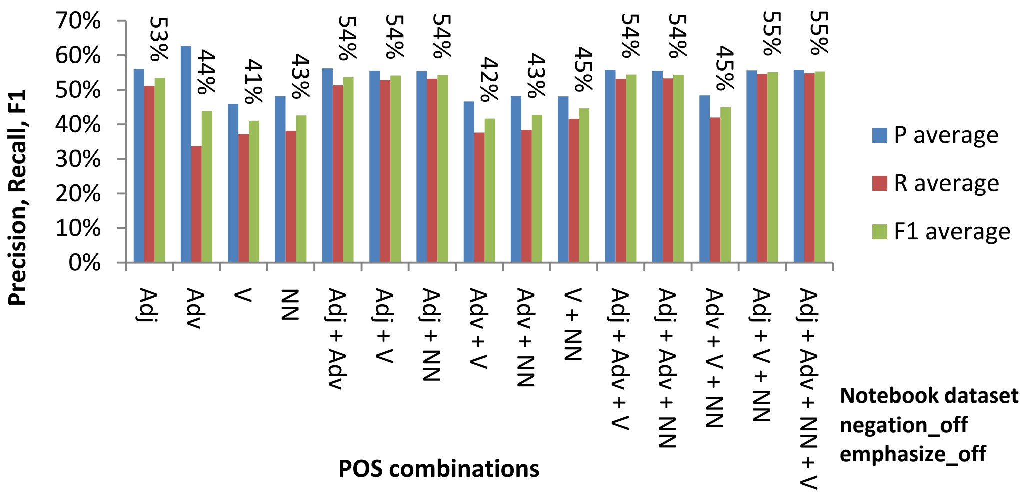


Figure 4.17: POS analysis using the Notebook dataset.

By the values provided in Figure 4.17 it is also confirmed that adjectives and their combinations with other POS are the best indicators of sentiment in sentences. The highest classification performance for the Notebook dataset is achieved when all POS are considered.

In this section it has been analyzed how different POS influence the classification results of Sentimal. In the following, the best configurations (combination of features and other settings) that produce the best classification results will be identified.

4.3.1.2 Identification of the best feature configuration for each dataset

In order to identify the optimal configuration options, 45 configurations have been tested. These configurations have been created by combining consideration of POS, of negation and of emphasizes. 15 configurations only consider POS combinations (see Section 4.3.1.1), 15 further ones consider POS combinations plus negation, and the last 15 configurations consider POS combinations plus negation plus emphasize.
In order to provide a more effective visualization, only the results of the best POS combinations are depicted for each of the datasets in the Figures 4.18 to 4.20.

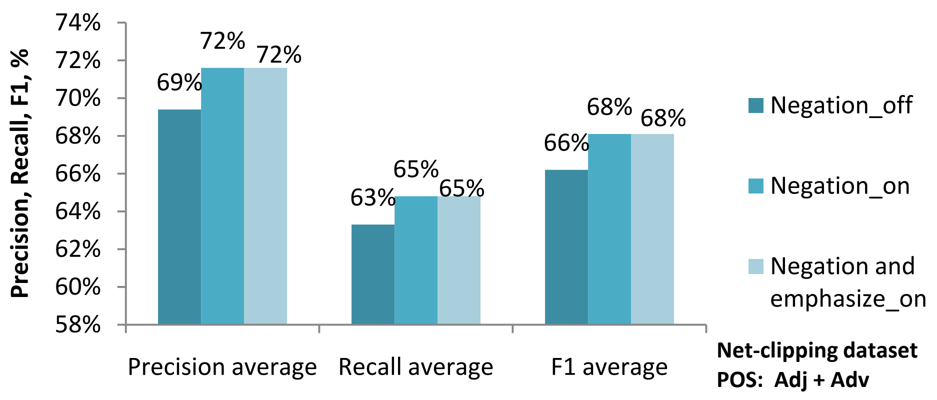


Figure 4.18: Testing influence of negation and emphasizing using Net-clipping dataset.

In Figure 4.18 we can see that if Sentimal's performance is tested using the Net-clipping dataset with considering negation, the value of the average precision is increased by 3%, the average recall is increased by 2%, and the average F1 score is also increased by 2%. On the other hand, considering the emphasize technique does not improve the classification performance. Due to this, the best configuration of Sentimal's feature set for Net-clipping dataset can be defined:

- POS: adjectives plus adverbs; considering negation (see Table 4.3).

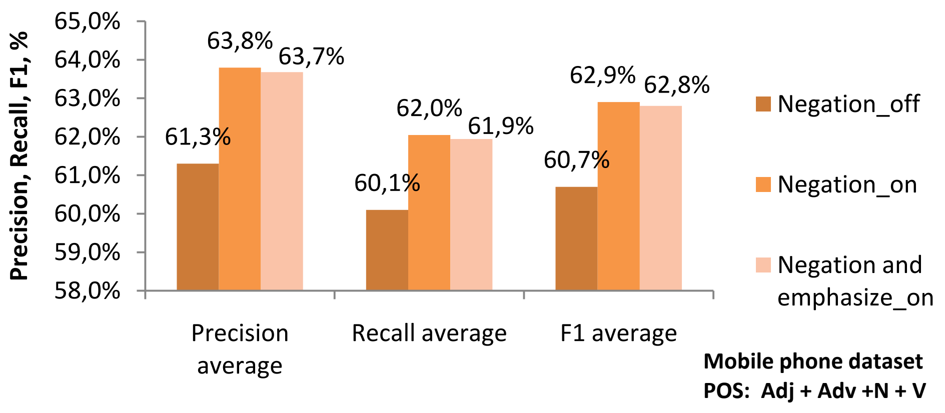


Figure 4.19: Testing influence of negation and emphasizing using Mobile phone dataset.

Figure 4.19 shows the comparison of Sentimal's classification results for the Mobile phone dataset when considering different settings. First not considering negation and emphasize, then considering negation, and in the last step, considering both negation and emphasize. For the best POS combination (when all four POS are considered) if negation is considered, the average precision increases by 2,5%, average recall raises by 2% , and average F1 score growth by 2%. Applying emphasizing technique does not improve classification performance but reduces the performance even slightly. As a result the best configuration of features for Sentimal using the Mobile phone dataset can be defined:

- POS: adjectives + adverbs + nouns + verbs and accounting for negation (see Table 4.3).

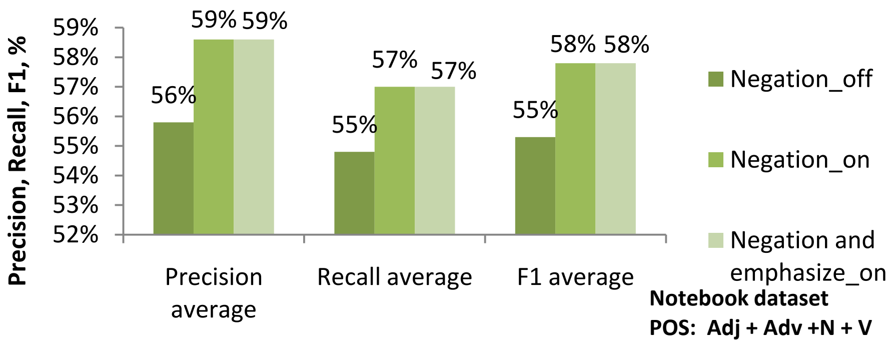


Figure 4.20: Testing influence of negation and emphasizing using Notebook dataset.

When testing these settings on the Notebook dataset, the results of average precision increased by 3% if negation is considered - the average recall is increased by 2%, and the average F1 score improved by 3% as well. This behavior can be seen in Figure 4.20. Considering emphasize technique does not influence the classification behavior. We can conclude that the best combination of features for Sentimal tested on Notebook dataset is:

- POS: adjectives + adverbs + nouns + verbs and accounting for negation (see Table 4.3).

The combinations of features for Sentimal for each dataset that results in the best classification performance are summarized on Table 4.3.

	Net-clipping dataset	Mobile phone dataset	Notebook dataset
POS	Adj + Adv	Adj + Adv + N + V	Adj + Adv + N + V
Considering negation	true	true	true
Considering emphasize	-	-	-

Table 4.3: The best configuration of features for Sentimal for each dataset.

4.3.1.3 Classification evaluation of three sentence tonalities

In the following, it will be discovered in more detail with which performance Sentimal classifies sentences of each tonality. It is important to find out if the classification results are skewed towards any tonality and then to analyze the reasons why this might be the case. Figure 4.21 presents Sentimal's classification results in metrics of precision, recall, and F1 for all sentence tonalities tested on the Net-clipping dataset for the best features combination (POS: adj +adv; considering negation).

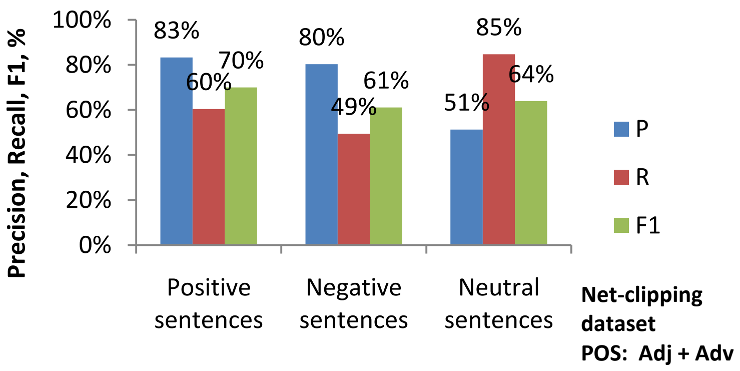


Figure 4.21: Evaluation of Sentimal's classification results of each sentence tonality using Net-clipping dataset.

Sentimal classifies positive sentences in the best way with a precision of 83%, a recall of 60%, and F1 of 70%, the neutral sentences show a high recall and low precision rate, and in the third place, negative sentences are classified with the lowest recall but the highest precision rate. For neutral sentences the metric of recall equals 85%, precision is 51%, and F1 equals 64%.

Figure 4.22 depicts Sentimal's classification results separating the three sentence tonalities (applied dataset: Mobile phone).

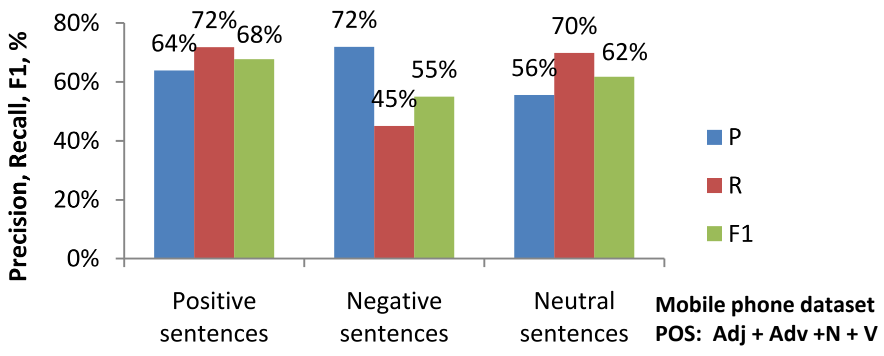


Figure 4.22: Evaluation of Sentimal's classification results of each sentence tonality using Mobile phone dataset.

By the results shown in Figure 4.22 it is confirmed that classification of positive sentences gives the best result, recognition of negative sentences has the same problem of low recall rate, and neutral sentences are classified with a high recall rate but lower precision rate.

Finally, the classification results for the three sentence tonalities using the Notebook dataset are depicted in Figure 4.23.

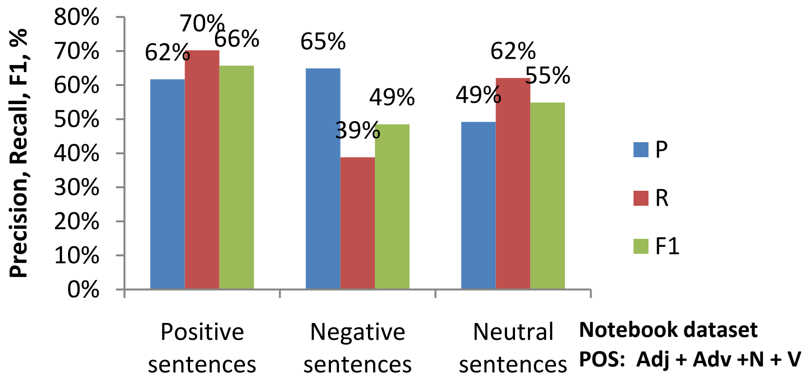


Figure 4.23: Evaluation of Sentimal's classification results of each sentence tonality using Notebook dataset.

In Figure 4.23 we can see the same tendency, that the positive sentences are classified in the best way due to the highest value of recall and as a results highest F1, then the neutral sentences, and in the worst case negative sentences.

The interpreted tendency in Sentimal's classification behavior for three datasets can be explained by analyzing the content of all three datasets in detail (See Section 4.5).

The main problems of classifying different sentence tonalities which arise when testing Sentimal on all three datasets are going to be addressed in the next section.

4.3.1.4 Analysis of influence of dataset peculiarities on tonality classification results

After analysing the tonality classification results of Sentimal two core problems appeared for each of the three datasets, which lower the general performance results:

1. a low recall rate (but at the same time high precision rate) for negative sentences and
2. a the low precision rate (but high recall rate) for neutral sentences.

The factors that influence recall rate of negative sentences in Sentimal are:

- Incompleteness of SentiWS vocabulary, namely the absence of quite some polar negative words in the dictionary. In this case, the words are considered to be neutral and it reduces the recall rate for negative sentences as well as precision rate of neutral sentences. As we can see the mentioned above two problems are interconnected.
- Incapability of classification algorithm to define negative sentences if they are formulated in a form of irony, using some stable expressions, or in subjunctive form (See Section 4.5).

The second problem is the low precision rate of neutral sentences. Sentimal's classification algorithm considers the word to be neutral if it is not present in SentiWS dictionary, and this assumption is generally correct. The only problem is that the dictionary is incomplete and thus does neither contain all sentiment words for German language nor for all domains.
It also can be stated that ambiguous words which have polarity in domain-specific datasets but are neutral for others as well as mistyping and misspelling mistakes hinder the Sentimal's classification performance.
The last step in evaluating the Sentimal's performance is to define if its classification results are domain independent.

4.3.1.5 Evaluating if Sentimal's performance is domain independent

As the following, evaluation of Sentimal's classification performance using all three datasets will be conducted and thus it will be analyzed whether Sentimal is a domain independent classifier. Figure 4.24 shows Sentimal's average classification results applying the best feature configurations for three datasets.

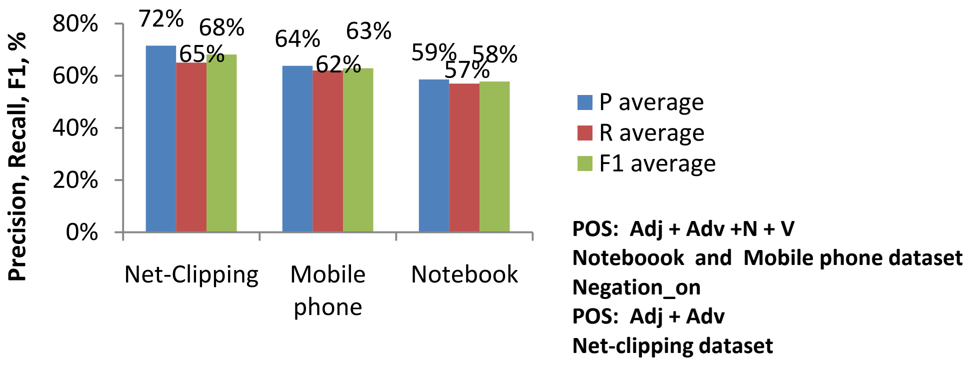


Figure 4.24: Comparison of Sentimal's performance tested on three datasets.

This figure proves that Sentimal classifies sentences of Net-clipping dataset most accurately with average precision of 72%, average recall of 65%, and average F1 of 68%. In the second place the sentences of Mobile phone dataset, and then of Notebook dataset are classified. The average value of F1 using Net-clipping dataset is 5% better than using Mobile phone and 10% better than when applying Notebook dataset. In Figure 4.25 Sentimal classification results for each tonality and each dataset are summarized. The best feature set for each dataset is applied (POS: adj plus adv; consider negation – when using Net-clipping dataset and POS: all POS, consider negation –when using Mobile phone and Notebook datasets).

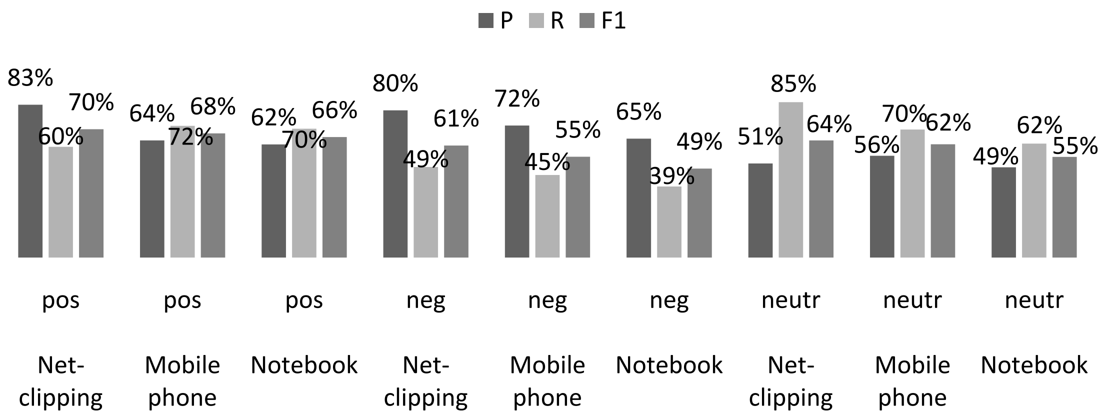


Figure 4.25: Comparison of Sentimal's classification of all sentence tonalities for all datasets.

As we can see Sentimal's classification results show that positive, negative, and neutral sentences are classified in the best way using Net-clipping dataset.

The positive sentences of Net-clipping dataset are classified with F1 of 70%, which is 2% and 4% higher than F1 values derived from classifying sentences of Mobile phone and Notebook datasets respectively. In general the difference cannot be considered too big and Sentimal classifies positive sentences of different domains well.

When analyzing and comparing the classification results for negative sentences we can see that the recall rate for all datasets is rather low. It can be explained by the fact that negative sentences in these datasets are often formulated in a form of irony, stable expressions or using subjunctive form. The examples of how negative sentences in these three datasets are constructed can be found in Section 4.5.

What about interdomain results of neutral sentence's classification we can observe the stated and analyzed in Subsection 4.3.1.4 problem, namely low precision rate when observing the high recall rate.

In general the same tendency in classification behavior of all sentence tonalities for each of three datasets can be observed. But the classification results of Sentimal using Net-clipping dataset are better than when considering Mobile phone, and Notebook datasets. It might be due to the peculiarities of these datasets (see Section 4.5). The gap in Sentimal's performance results might be covered when on the one hand extending SentiWS dictionary with domain specific words from Mobile phone and Notebook domains, and on the other hand developing some new sophisticated approaches in order to recognize irony in the sentences or subjunctive form of formulating negative and positive sentences.

4.3.1.6 Summary of Sentimal evaluation results

The evaluation results showed that in German language adjectives and their combinations with other POS are the best indicators of sentiment in the sentences. Adding other POS to adjectives improve the classification performance in metrics of average F1 score from one to two percents. But as we can see the difference is not too big.

The best configurations of feature set for each of three datasets were identified. When using Net-clipping dataset the following features should be considered

- POS: adjectives and adverbs; accounting for negation.

When using Mobile phone and Notebook datasets

- POS: combination of all POS and accounting for negation.

Emphasizing technique does not improve the classification results. Negation recognition improves classification performance up to 3%. It is a good indicator when taking into account that about 5 % of sentences in datasets are negated using negation words.
In general Sentimal classifies best of all sentences of positive tonality, in the second place neutral and in the worst case negative sentences.
Even though Sentimal shows good classification results for classifying positive sentences of datasets from all domains, the average classification results are the best using the Net-clipping dataset. The improvement of classification results for Mobile phone and Notebook dataset may be reached by extending SentiWS dictionary with domain specific words and developing further algorithms for analyzing sentences in subjunctive form.

4.3.2 PTC features analysis

In this chapter the evaluation of PTC classification results applying considered set of features, namely word-level n-grams and character level n-grams will be done. Apart from n-grams, the influence of applying stopwords removing and stemming on PTC classification performance will be estimated. The classifier will be learned using Net-clipping, Mobile phone and Notebook datasets and it will be checked how good PTC classifies each sentence tonality and whether its classification performance is domain independent.

4.3.2.1 PTC word n-grams vs. char n-grams

In general 144 configuration runs were created for testing PTC performance on each dataset applying word-level n-grams plus considering preprocessing techniques. And another 160 configuration runs for character-level n-grams and considering preprocessing techniques.

In the first step the classification results of PTC tested on Net-clipping dataset, applying char-n-grams and preprocessing techniques will be evaluated. The first 40 configuration runs considered char n-grams, another 40 ones considered char n-grams plus stemming, the next 40 configuration runs considered char n-grams and stopwords removing, and the last 40 ones considered char n-grams and both stemming, and stopwords removing. Figure 4.30. shows PTC classification results of several configuration runs.

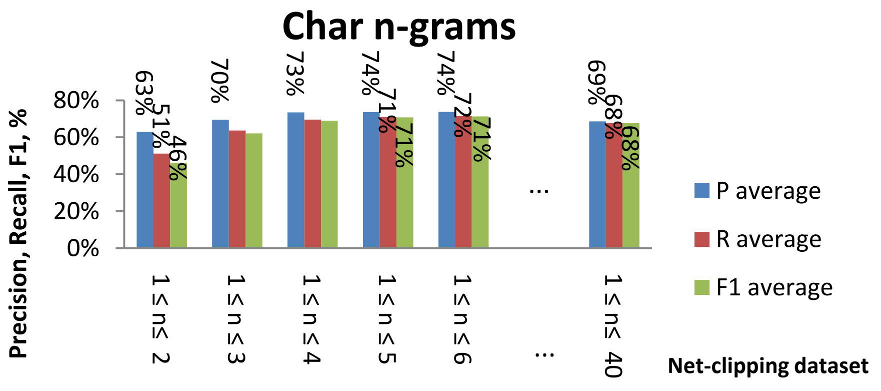


Figure 4.26: PTC classification performance using Net-clipping dataset for char n-grams.

In this figure PTC performance applying the first five and the last one (1 ≤ n ≤ 40) configuration runs can be seen. Using Net-clipping dataset and char n-grams (1 ≤ n ≤ 6), not considering stemming and stopwords showed the best classification results with precision = 74%, recall = 72%, and F1 score = 71%. After implementing other 120 configuration runs and evaluating the results it was discovered that stemming and stopwords removing does not improve PTC's classification performance.

As following the best configuration of features for PTC using Net-clipping dataset can be defined:

- Char n-grams (1 ≤ n ≤ 6); considering neither stemming nor stopwords removing (see Table 4.4).

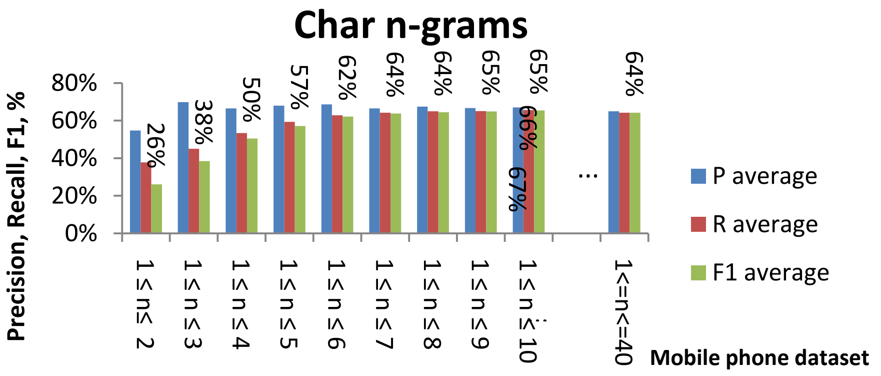


Figure 4.27: PTC classification performance using Mobile phone dataset for char n-grams.

Figure 4.27 shows PTC classification results for the first 40 configuration runs using Mobile phone dataset. It proves that the best feature combination testing PTC on Mobile phone dataset is char n-grams (1 ≤ n ≤ 10) without considering preprocessing techniques. PTC classification results for the next 120 classification runs (when considering stemming and stopwords removing) almost do not influence classification performance. For some features the performance improves or decreases but even less than for 0,5%. That is why we can argue that considering preprocessing techniques does not influence classification results. In the following the best configuration of features for PTC using Mobile phone dataset is defined:

- Char n-grams (1 ≤ n ≤ 10); considering neither stemming nor stopwords removing (see Table 4.4).

As the last step 40 configuration runs were applied using Notebook dataset which is depicted in Figure 4.28.

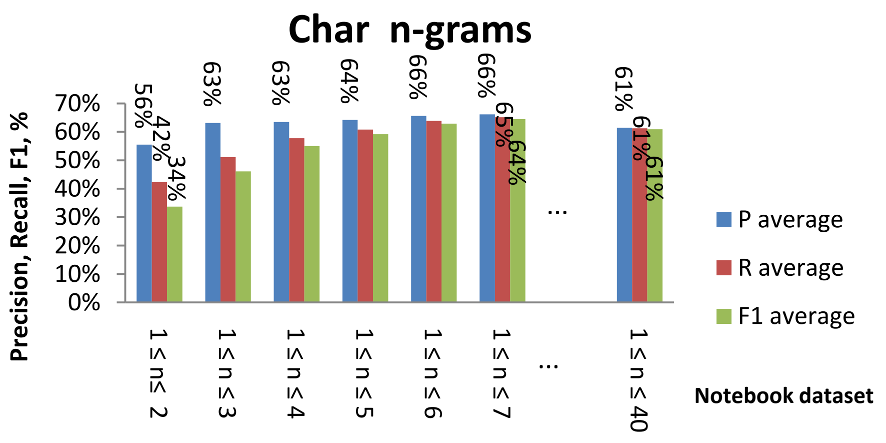


Figure 4.28: PTC classification performance using Notebook dataset for char n-grams.

Testing PTC on Notebook dataset showed the best results (precision of 66%, recall of 65%, and F1 of 64%) for char n-grams (1 ≤ n ≤ 7) without considering stemming and stopwords removing. As the following the best configuration of the feature set for PTC using Notebook dataset is:

- Char n-grams (1 ≤ n ≤ 7); considering neither stemming nor stopwords removing (see Table 4.4).

As the following, word-level n-grams will be analyzed.

For testing PTC on all defined in experimental setup word-level n-grams another 144 configuration runs were realized. Applying word level n-grams in comparison to char-level n-grams resulted in worse PTC performance, using all three datasets. Figure 4.29 proves this statement and depicts the results for the first seven classification runs (Used dataset: Mobile phone).

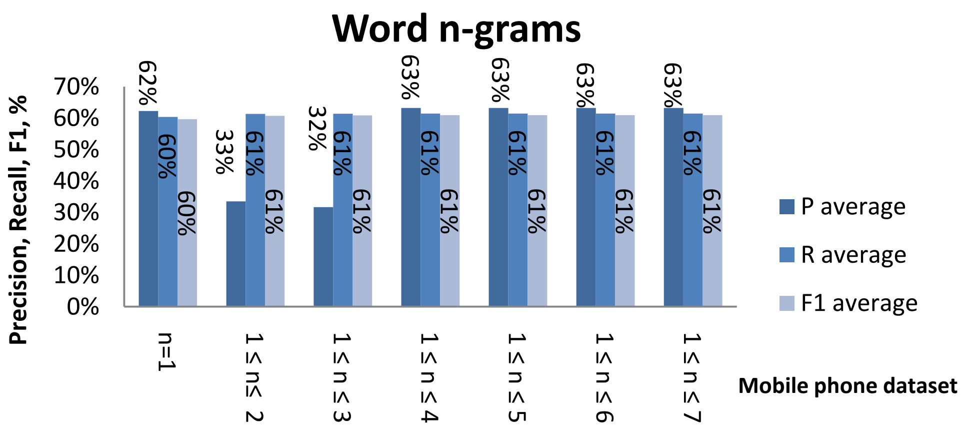


Figure 4.29: PTC classification results for word-level n-grams.

From the figure it can be seen that increasing the number of word n-grams beyond four does not improve the classification performance anymore. This tendency is also the same for Net-clipping and Notebook datasets. It means that it is unlikely that five or six the same coincide words will appear twice in any of the testing sentences.

When testing PTC performance using higher level word n-grams and applying all three datasets it became evident that the performance decreases when increasing the number of n. The general tendency of PTC performance which is applicable when using all three datasets is depicted in Figure 4.30 on the example of Net-clipping dataset.

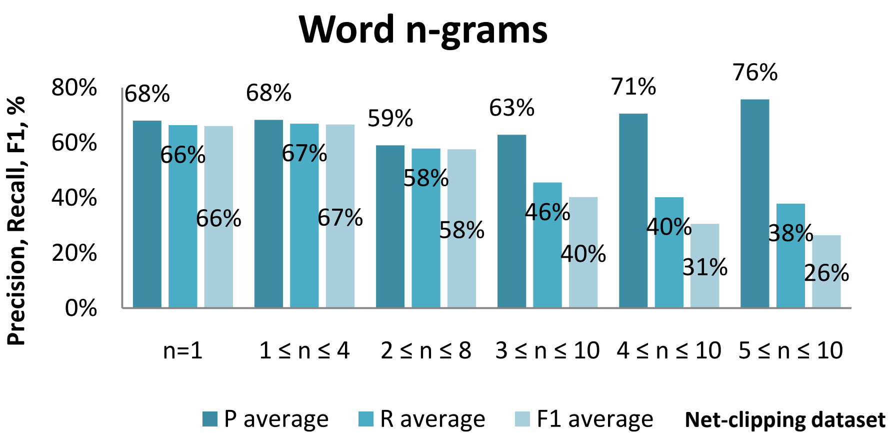


Figure 4.30: PTC classification results for higher-level word n-grams.

Comparing the performance of PTC using word and char n-grams we can conclude that using word-level n-grams show worse performance in metrics of precision, recall, and F1. Applying char-level n-grams resulted in determining the best configuration of features for PTC. Figure 4.31 proves mentioned above statement. It can be seen that precision and F1 rate of the best word n-gram combination (1 ≤ n ≤ 4) is four percent lower than the precision rate for the best char n-gram combination (1 ≤ n ≤ 10) when using Mobile phone dataset.

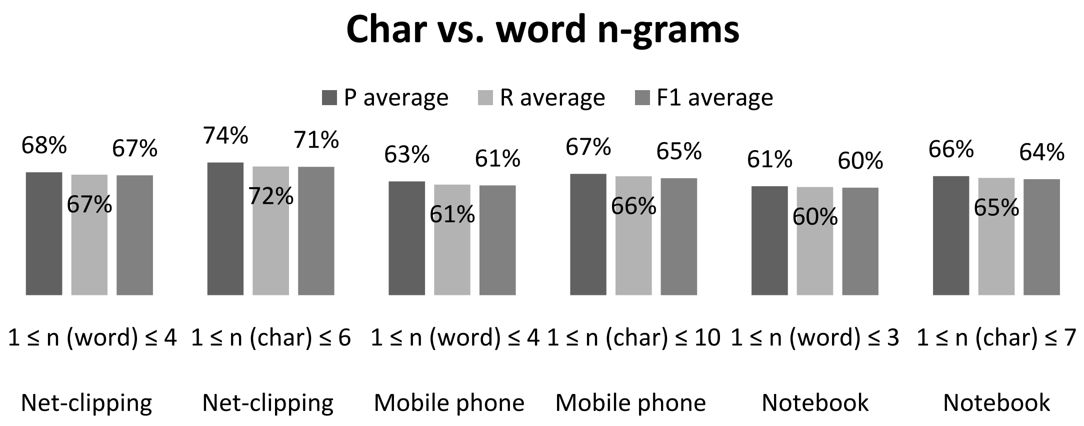


Figure 4.31: Char n-grams vs. word n-grams.

The results of the best configuration of features for PTC for the three datasets are depicted on Table 4.4. The results were derived when training and testing PTC classifier with 70% of training and 30% of testing data.

Features and preprocessing	Net-clipping dataset	Mobile phone dataset	Notebook dataset
Word n-grams	-	-	-
Char n-grams	1 ≤ n ≤ 6	1 ≤ n ≤ 10	1 ≤ n ≤ 7
Removing stopwords	-	-	-
Stemming	-	-	-

Table 4.4: The best configuration of features for PTC for each dataset.

4.3.2.2 PTC classification results of three sentence tonalities

In this section it will be analyzed how PTC classifies each of the sentence tonalities. In order to do this the classification results for the best configurations of features were analyzed and visualized. Figure 4.32 presents PTC classification results of three sentence tonalities tested on Net-clipping dataset.

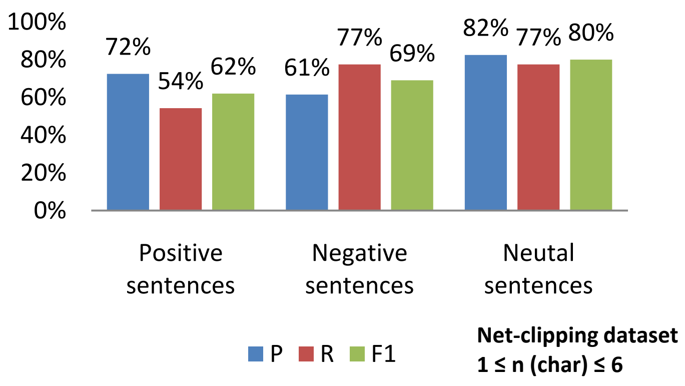


Figure 4.32: PTC classification results of three sentences tonalities using Net-clipping dataset.

From this figure we can derive that using sentences of Net-clipping dataset PTC classifies in the best way neutral sentences with the best precision = 82%, recall =77%, and F1 score = 80%, then negative sentences, and in the worst case positive sentences, due to the fact that the value of recall is the worst, only 54%.

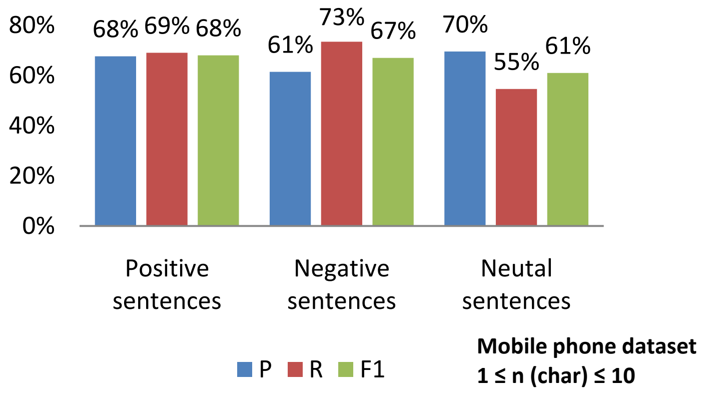


Figure 4.33: PTC classification results of three sentences tonalities using Mobile phone dataset.

For Mobile phone dataset the results look different, here PTC shows the best performance for classifying positive sentences (precision =68%, recall =69%, and F1 = 68%), then negative, and in the worst case neutral sentences. It can be seen in Figure 4.33.

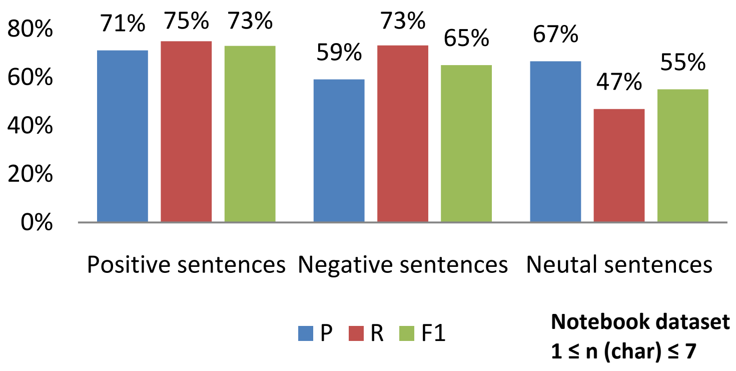


Figure 4.34: PTC classification results of three sentences tonalities using Notebook dataset.

PTC classification results of three sentence tonality using Notebook dataset have the same tendency like using Mobile phone dataset, see Figure 4.34. It classifies with the best performance positive sentences, then negative, and in the worst case neutral sentences with the value of F1 equal 73%, 65%, and 55% correspondently.

4.3.2.3 Checking if PTC is domain independent classifier

In order to define whether the results of PTC are domain independent, let us consider the data presented in Figure 4.35, in which the average performance results (averaged results of three sentence tonalities) are shown. The classifier is tested applying the best combination of features defined for each of three datasets (see Table 4.4).

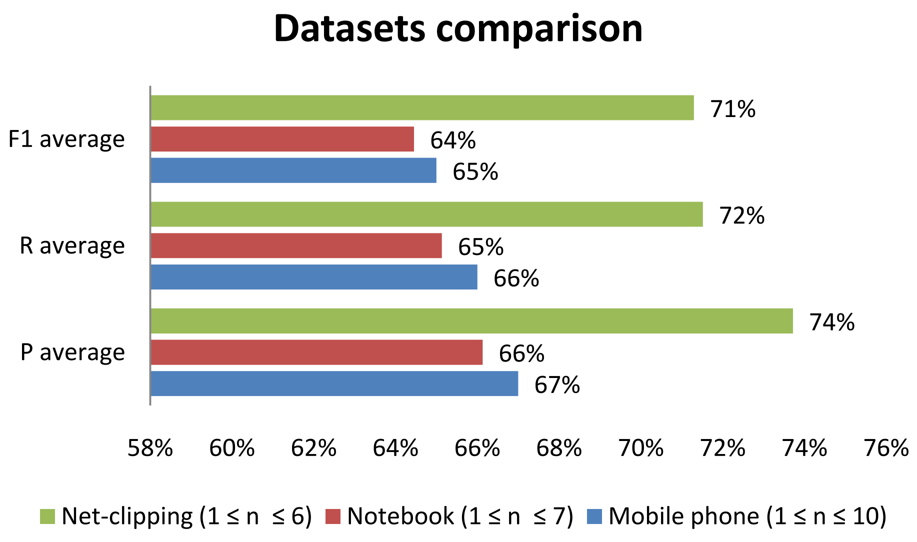


Figure 4.35: PTC average performance - comparison of three datasets.

We can see that average classification results of PTC using Net-clipping dataset are apparently better than using Mobile phone, and Notebook datasets. With average precision seven percents higher and average recall six percent higher than for Mobile phone dataset. The average results are higher for

Net-clipping dataset due to the fact that neutral sentences in Net-clipping are classified much better than in Notebook and Mobile phone datasets. Positive and negative sentences of Net-clipping dataset are classified even a bit worse than from Notebook and Mobile phone datasets (see Figures 4.32 to 4.34).

The classification skewing of PTC towards Net-clipping dataset can be explained with that fact that the sentences in Net-clipping dataset were collected within three days time interval from the blog posts, news portals, chatting portals etc., where three main topics were at the centre of attention. As a result the sentences contain a lot of the same words, which results in extra similarity of training and testing datasets. And this is not the case in Mobile phone and Notebook datasets. That is why using Net-clipping dataset result in better classification performance of PTC. After analyzing the content of Net-clipping dataset it can be considered that if Net-clipping dataset would be composed of the topics collected within the monthly time interval there would be less similarity in the training and testing data and the results would be more similar to the Notebook and Mobile phone datasets.

4.3.2.4 Summary of PTC evaluation results

Performance of PTC using a vast set of features was evaluated. It was discovered that when using char-level n-grams PTC performance is better than when considering word-level n-grams. The best combination of features was defined for each of the datasets. They are char level n-grams with $1 \leq n \leq 6$ for Net-clipping dataset, $1 \leq n \leq 10$ for Mobile phone, and $1 \leq n \leq 7$ Notebook datasets. Such preprocessing techniques as stemming and stopwords removing do not make a positive influence on classification performance.

It was discovered that PTC classifies neutral sentences in the best way, then negative, and afterwards positive sentences of all three datasets.

PTC classified neutral sentences of Net-clipping dataset much better than of Mobile phone and Net-clipping dataset which resulted in a higher average classification results. It may be due to the fact that Net-clipping data was collected within three days, which result in the similarity of training and testing data, which in its case improves the performance results.

4.4 Comparing PTC and Sentimal

In order to provide a fair comparison of PTC and Sentimal performance, these classifiers were tested on the same testing dataset (the same 30% of testing data used for PTC). For comparison only the best configuration of features for PTC and Sentimal was considered. PTC classification results using each of three dataset are presented in Figures 4.36 to 4.38.

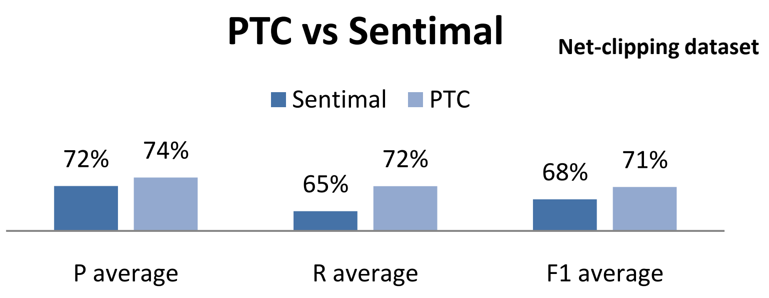


Figure 4.36: Comparing two classifiers performance using Net-clipping dataset.

We can see that PTC shows better performance than Sentimal. PTC results in metrics of precision recall, and F1 are better than of Sentimal. Precision is 2% higher, recall 7% higher, and F1 3% higher.

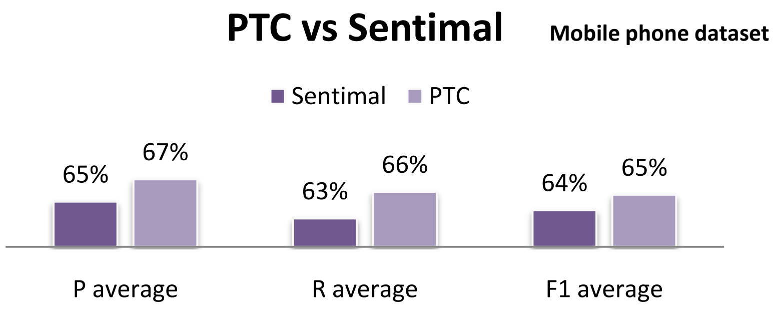


Figure 4.37: Comparing two classifiers performance using Mobile phone dataset.

Using Mobile phone dataset the average performance of PTC in metrics of F1 appeared to be only 1% better than of Sentimal.

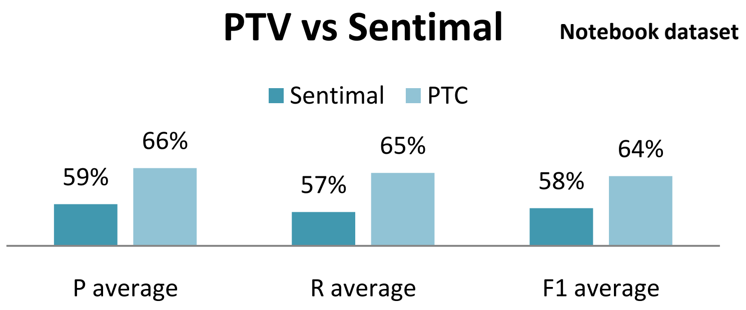


Figure 4.38: Comparing two classifiers performance using Notebook dataset.

Sentences from Notebook domain are much better classified by PTC than by Sentimal. With 7% increment in precision, 8% increment in recall and 6% increment in average value of F1.

Figure 4.39 draws the average classification results for all three datasets.

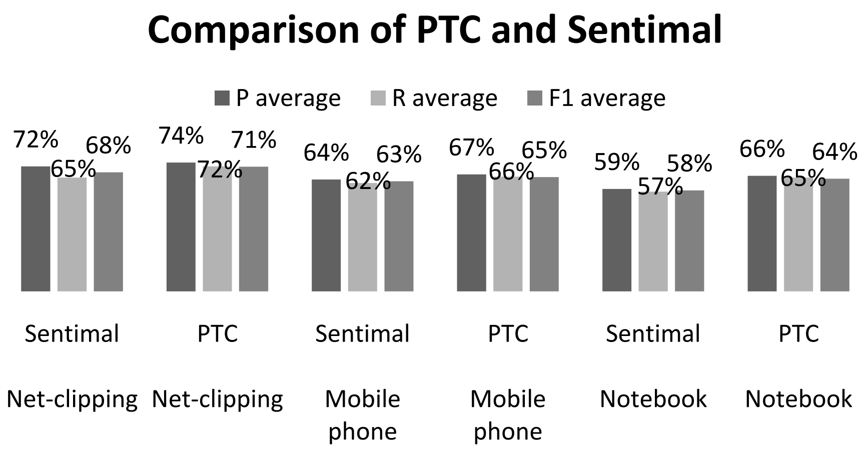


Figure 4.39: Comparing two classifiers performance on three datasets.

Although PTC shows better classification results than Sentimal, the performance results shown by Sentimal should nevertheless be regarded as promising. Its classification performance in metrics of F1 is only 1% lower using Mobile phone dataset and 3% lower using Net-clipping dataset. It is believed that with extension of SentiWS dictionary and further extension of Sentimal possibilities e.g. classification of sentences formulated in subjunctive form will further improve the Sentimal's performance.

4.5 Content analysis of Net-clipping, Mobile phone, and Notebook datasets

As the classification results of the classifiers vary considerably, particularly dependent upon on the type of dataset used, it is important to analyze the content of all three datasets. A short analysis will be presented in the following in order to enable the reader to understand the mentioned classifier behaviour. We will focus on two core aspects: Firstly, the style of the language used in datasets and, secondly, the way negation in sentences is expressed will be discussed.
In these regards, several questions are of crucial importance: Is it the case that mainly domain-specific words occur in the datasets? Do people use the formal language or slang words to express their thoughts? Do articles in some Internet resources contain more spelling mistakes? How are the negation sentences constructed i.e. is it a normal form of expressing negation (via using negation words such as *nicht, nie, kein, keine*...), or is negation expressed in a form of irony, or via some stable expressions like *"alles andere als das Gelbe von Ei sein",* or via subjunctive constructions?
All these aspects are important to consider, as they have an obvious influence on the classification behaviour of classifiers and analyzing these peculiarities will help to explain the classification results.
After manually analysing all three datasets and carefully reading through all the sentences it became evident that the articles in Net-clipping dataset contain more formal language than articles in the Notebook and Mobile phone datasets. This phenomenon can possibly be explained by the fact that these articles are derived from blog posts, news portal, Wikipedia etc. The articles from Mobile phone and Notebook datasets are derived from Amazon product reviews and thus contain a more frequent use of slang words. In general, there are more mistyping and misspelling mistakes which result in the lower recall during the classification step. Another point is that most of the slang words are not present in the SentiWS dictionary, which may result in the lower precision and recall rate e.g. one of such sentences *"Lediglich das WLAN wollte erst nicht **klappen**".* The word "*klappen"* is not present in the SentiWS dictionary and will be classified by Sentimal's classification algorithm as neutral word which will lead to the classification mistake and thus we will get a lower precision rate for neutral sentences and lower recall rate for negative sentences.

Negation of sentences appears to be constructed in these datasets in a different way as well. Due to the nature of Internet resources used for datasets creation sentences in news-portals, Wikipedia or blog-posts are negated mainly using negation words while in Amazon product reviews people like to describe things in a casual way, often using irony when the product does not meet the customers' expectations, or some special expressions which could only be understood by humans. In all three datasets negation is often expressed in a subjunctive form *„Es wäre sehr schön gewesen, wenn wie angeben war das maximum 4GB aufzurüsten ginge.“* which is not detectable by Sentimal's classification algorithm.
Appendix B shows selected examples for expressing negation in Net-clipping, Notebook and Mobile phone dataset and for the ironic style of expressing negation, which is often used in the Mobile phone and Notebook datasets.

Moreover, in Net-clipping, Notebook, and Mobile phone datasets the subjunctive constructions are often used in order to express wished but unreal condition. It makes these sentences difficult to be analysed by automatic means and this in its turn results in lower precision and recall rate. One example of such an expression is:

„Man hätte zumindest eine Recovery-DVD, also einen Datenträger mit dem sich Windows und Treiber wieder aufspielen lassen, dazu legen können.“ (Notebook dataset).

In such sentences the classification algorithms of Sentimal would collect the positive sentiment words, and analyse the sentence by mistake as positive, unaware of that, that the unreal condition is described in the sentence.

Ambiguous words lower the classification precision as well, as these words appear to be polar in domain specific datasets, but may carry no sentiment in other sentences. For example consider the following sentence from the notebook domain:

"Natürlich ist mir dann auch gleich aufgefallen, wie laut das DVD-Laufwerk beim DVD-brennen rattert.“ or another sentence *"Das DVD Laufwerk ist recht laut wenn es anspringt und voll durchläuft.“* (Notebook dataset).

In this sentence, the term "*laut"* has negative polarity indicating that the device produces a loud noise when doing some work and the sentence receives a negative tonality as well. On the other hand, the term "*laut"* may have no sentiment. This is e.g. the case in the following example:

„Laut einer Forschung, ist das Unternehmen „A“ der Leader im Markt“.

Summarizing all these aspects mentioned above it is important to admit that the datasets' peculiarities i.e. the language style used, the way of building negation, orthography, mistyping, ambiguous words and using subjunctions in sentences are the factors resulting in varying precision and recall values.

As a side-result of this analysis, a small list of words has been created from Notebook and Mobile phone datasets that have a polarity and often appear in the datasets but are not present in the SentiWS dictionary. It is shown in Appendix C. Adding these words to the dataset might be useful for those who will be working on extending SentiWS dictionary or doing sentiment analysis of data from Notebook or Mobile phone domains.

4.6 Conclusion

In this chapter a detailed analysis of two sentiment classifiers applied to the German language was provided.

As a preliminary step for this analysis, a detailed description of three datasets applied in this master thesis and central characteristics of these datasets were discussed in Sections 4.1 and 4.2:

- It was pointed out, that all datasets should contain the same amount of sentences of three tonalities in order to provide fair classification results which are not influenced by one of the dominating sentence tonality.
- Furthermore, it was experimentally proved that the available amount of sentences in the datasets is sufficient for further research.
- The experiments for defining the optimal percentage of training and testing data for learning PTC were realized. It was shown that with increasing the amount of training data the results of PTC performance improved. PTC shows quite good performance when 70, 80, and 90 percentages of data were applied for training the classifier. The best performance achieved was when 90% of data is used for training, but 10% left for testing were not enough to reflect

the classification results of the complete dataset. That is why statistical tools were used and the minimum amounts of sentences which need to be left for testing the PTC classifier were identified. The results showed that at least 30% of data from each dataset should be used for testing and 70% for training the classifier which are the optimal learning characteristics for PTC.

Section 4.3 addresses the main research questions of this master thesis and evaluates the results, which are briefly summarized in the following.

First of all, the classifiers were tested on the considered set of features, the influence of these features on classification performance were analysed, the best classifier configurations identified and it was checked whether the classifiers results are domain independent.

- Sentimal uses POS as features and it was proved that in German language adjectives are the best indicators of sentiment in sentences. Thus, the core feature used for sentiment detection does not differ from the one used e.g. in classifiers focussing on the English language. Besides this evident result, the concrete performance of applying these features was analysed. Their combination with other POS improves the classification performance in metrics of F1 by 2%. Considering negation improves the value of F1 by 2%, but considering emphasize does not induce positive influence on classification results. All results mentioned above are true when testing the classifier on three datasets.
- The best set of features for Sentimal for three datasets are defined in the following:
  - POS: adjectives plus adverbs; consider negation (Net-clipping dataset).
  - POS: adjectives plus adverbs, plus nouns, plus verbs; consider negation (Notebook dataset).
  - POS: adjectives plus adverbs, plus nouns, plus verbs; consider negation (Mobile phone dataset).

It was checked if Sentimal's results are skewed towards a particular sentence tonality and if it is a domain independent classifier. The results of this analysis are summarized in the following:

- Sentimal classifier shows the same classification tendency when applying each of three datasets. It gives the best results for positive sentences, then neutral, and then negative sentences.
- Sentimal shows better classification results when using the Net-clipping dataset, then using the Mobile phone and Notebook datasets.

This classification tendency can be explained by the peculiarities of the datasets and Sentimal's classification algorithm and are in detail analyzed in Subsection 4.3.1.4 and Section 4.5.

The central research results for PTC are presented in the following.

- Among the features considered for PTC, character level n-grams resulted in better performance than when considering word-level n-grams.
- Considering the preprocessing techniques like stemming and stopwords removing does not make positive influence on PTC classification performance.
- The best configurations of PTC using each of the three datasets are presented in the following.

    - Char n-grams (1 ≤ n ≤ 6); considering neither stemming nor stopwords removing (Net-clipping dataset).
    - Char n-grams (1 ≤ n ≤ 10); considering neither stemming nor stopwords removing (Mobile phone dataset).
    - Char n-grams (1 ≤ n ≤ 7); considering neither stemming nor stopwords removing (Notebook dataset).

- PTC gives the best results for classifying positive, then negative, and afterwards neutral tonalities when applying Notebook and Mobile phone datasets, but when using Net-clipping dataset classification of neutral sentences is the most accurate (with precision rate of 82% and recall rate of 77%), the second well are classified negative sentences (with high recall rate of 77% and precision rate of 61%), and then positive sentences with high precision of 72% but relatively low recall only of 54%.
- PTC classifies sentences of Net-clipping dataset more precise (with average precision equal 74%) than in the case of the Mobile phone and Notebook datasets (average precision equal 67% and 66% respectively). PTC's classification results are biased towards Net-clipping dataset as this dataset was collected within three days, which results in bigger similarity of training and testing data, and thus better classification results.

In Section 4.4 it was concluded that the best configuration of PTC showed better classification performance tested on all three datasets than Sentimal. But the performance of Sentimal is also promising and can be improved by both extending the SentiWS dictionary, which is applied for analysis, and by adding new functionality to the classification algorithm, such as identifying sentences formulated in subjunctive form, or defining irony.

In Section 4.5 the content analysis the datasets used in this master thesis, namely Net-clipping, Mobile phone and Notebook dataset is presented, which helps to analyze the classification results of both classifiers.

Chapter 5: Epilogue

5.1 Summary

As there exist no detailed analysis about classification performance of German sentiment classifiers this master thesis has an intention to close this gap and suggest classification algorithms and approaches which can be used for German language document's tonality classification. Thus, two opinion classifiers for German documents, the lexicon-based Sentimal and the supervised learning based approach PTC, have been analysed and compared as well as developed preprocessing and classification algorithms embedded in them.

Chapter 1 motivated the research work, outlined the research questions and provided the general structure of the complete document.

In Chapter 2 the most important background information has been presented. It includes an overview of lexicon-based approaches for sentiment classification and those, based on supervised learning. While the results of this work intend to be embedded into any German language sentiment classification system, the state of the art analysis of existing academic systems for sentiment analysis as well as employed algorithms of sentiment classification in them have been introduced. Unfortunately, only systems which classify English language documents were analysed as no published results for German language sentiment classification systems have been found. Afterwards, a list of existing commercial systems for analysing sentiment of multiple languages including German language are presented but they do not provide any implementation details of the employed algorithms.

Chapter 3 introduced the principles of PTC and Sentimal's work and described two preprocessing and opinion classification algorithms developed and employed. The chapter presented the experimental setups for both classifiers. Furthermore, implementation details of both classifiers' performance have been discussed.

In Chapter 4 a detailed description of the datasets applied in this thesis has been provided. Afterwards, the extensive evaluation of classifiers, using different features has been presented. As a result the best configuration of features for each classifier has been determined and the evaluation results of determining three sentence's tonalities have been presented. Every classifier has been checked whether its classification results are domain independent and the classifier which shows better classification results has been identified.

After this overview of the steps taken in this thesis, the research question forming the starting point of this thesis are discussed and answered. Furthermore, an outlook on further work is provided.

5.2 Research questions

In this section the answers to the research questions which were addressed by this master thesis are discussed.

- **Question 1**: Is the lexicon-based Palladian Sentiment Classifier better than Palladian Text Classifier based on supervised-learning?

In order to answer this question, a detailed analysis of two classifiers performances was conducted applying considered features set. Thus, the best configurations of features which result in the best performance metrics were identified for both classifiers. In order to determine which classifier performs best, PTC and Sentimal were tested on the same testing datasets derived from three considered datasets, namely Net-clipping, Mobile phone, and Notebook. The best configurations of features were applied to these classifiers. After evaluating the results it becomes evident that tested on all three considered datasets PTC classifier which is based on a supervised learning approach shows better performance results than lexicon-based Sentimal. Though PTC's and Sentimal's results in metrics of average precision (averaged over all three tonalities) are only slightly different, when applying the Mobile phone and Net-clipping datasets, but the average recall values produced by PTC are quite higher using these datasets (3% higher when applying mobile phone dataset and 7% higher when using the Net-clipping dataset). The average precision showed by PTC using the Notebook dataset is 7% higher, average recall 8% higher and average F1 6% higher than respective results of Sentimal. See Section 4.4 for more information. Nevertheless, Sentimal's performance is also quite promising and might be improved by extending the dictionary with domain specific words and adding new functionality to Sentimal's classification algorithm.

- **Question 2**: Which features should be applied to each classifier in order to achieve the best classification performance?

The best configuration of features was identified for each of the classifiers. In lexicon-based Sentimal classifier all parts of speech were analyzed concerning their influence on the total sentence sentiment. Adjectives and their combinations with other POS of German language showed the best classification results. Considering negations improved the classification results for 2-3% which is quite good result if to notice that the negated sentences (applying negation words) constitute less than 5% of all sentences in datasets. Considering emphasize does not influence classification results of Sentimal. For PTC character-level n-grams showed better performance than word-level n-grams. The preprocessing techniques such as stemming and stopwords removing do not improve the classification performance. The best configuration of features for Sentimal and PTC tested on three datasets are presented in Table 4.3 and in Table 4.4.

- **Question 3**: Are the classifiers' performances skewed towards a particular sentence tonality?

Yes. The classification results showed that in general both classifiers assort better positive sentences. Due to the difference of classification algorithms applied, the tendency of tonality analysis for each classifier differs as well. Sentimal classifies in the best way positive sentences with a high precision and recall rates (applying Net-clipping dataset precision equals 83%, recall 60%; applying Mobile phone dataset precision equals 64%, recall 72%, and using Notebook dataset precision is equal 62% and recall 70%), then neutral, and lastly negative sentences (see Figure 4.25). Such classification tendency, tested on all three considered datasets can be explained by peculiarities of these datasets. For more information see Section 4.5.
PTC tested on Mobile phone and Notebook dataset gives the best results for classification of positive sentences, then negative and afterwards neutral. Being tested on Net-clipping dataset PTC provides the most accurate results for neutral sentences (with the highest precision of 82% and recall of 77%),

then negative (with quite high recall of 77% but lower precision rate of 61%), and afterwards positive sentences (with precision equal 72% and recall 54%). For more details see Subsection 4.3.2.2 and Subsection 4.3.1.3.

- **<u>Question 4:</u>** Is PTC's and PSC's (Sentimal's) classification behavior domain specific?

Both PTC and Sentimal show better performance using the Net-clipping dataset in comparison to the other datasets. Sentimal's classification results in metrics of average F1 tested on Net-clipping dataset give 5% higher results in comparison to the tests performed using the Mobile phone dataset, and 10% higher in comparison to applying the Notebook dataset (see Figure 4.24).
PTC produces similar classification results if it is tested on Mobile phone and Notebook datasets but when applying Net-clipping dataset the performance in metrics of average F1 is 6% higher in comparison to Mobile phone dataset and 7% higher comparing to Notebook dataset (See Figure 4.35).
Thus, the results of both classifiers depend on the applied datasets and they are domain specific.
Adding functionality to the Sentimal's classification algorithm and extending the SentiWS dictionary can improve the inter-domain classification performance of Sentimal. As well as experimenting with the Net-clipping dataset (collecting data within a monthly period instead of three days, which would lead to the less similarity of the training and testing data) could result in similar inter-domain classification results of PTC. Thus, these are some of the tasks for future research work.

5.3 Future research work

After realizing the experiments and analysing the results the following further questions arise:

- How can recall value of negative sentences be improved in the case of the Sentimal classifier?

It was discovered that classification of negative sentences was realized by Sentimal in the worst case, due to the low recall value, applying three considered datasets. In order to increase the recall value of negative sentences it was analysed how negative sentences are formulated in these datasets, namely via negation words, using adjectives of negative tonality, using subjunctive form, irony, or expressed as an idiom. The last three aspects of negation formulation require attention as they are not solved by existing Sentimal's classification algorithm. The classification algorithm is flexible and can be extended by additional functionality. For example a new algorithm for subjunctive form identification can be developed, or a list of idioms for expressing negations can be collected and used by the algorithm for identifying negative idiom sentences. That might increase the recall and precision values of classifying sentences with negative tonality.

- How to make Sentimal's classification performance more domain independent?

To answer this question, further work on extending SentiWS dictionary, used by Sentimal classification algorithm with lexicon words from different domains should be done. This task requires huge effort, as all ambiguous words which have a tonality for one context but are neutral in another should be eliminated.

- How to improve Sentimal’s classification results?

It might be helpful to try out different word scoring algorithms, in order to assign a sentiment weight to each sentiment word. So far, Sentimal uses the weights of sentiment words which are available in SentiWS. But it was proved by (Cesarano, 2006) that the same algorithm produces different classification results when applying different word scoring mechanisms. Trying some of the available algorithms or developing a new one might be beneficial for general classification results.

- Do further supervised learning techniques perform better than dictionary-based approach?

Due to the fact that the classification results of PTC outperform those of Sentimal, it will be interesting to try out other supervised learning approaches, such as KNN, Naive Bayesian or SVM classifiers and test their performances applying the datasets considered in this master thesis.

Appendix A

The list of emphasize words, which were used in master thesis together with corresponding emphasize weight.

Emphasize word	Emphasize weight
sehr	2,0
deutlich	2,0
unheimlich	3,0
absolut	3,0
vollkommen	3,0
extrem	3,0
besonderes	3,0
extra	2,0
bisschen	0,9

Table A.1: List of emphasize words.

Appendix B

Some examples of formulating negative sentences in Net-clipping, Mobile phone, and Notebook datasets.

„Bisher erreicht nicht genug Hilfe für die notleidenden Menschen". (Net-clipping dataset).
In the following some examples of the ironic style of expressing negation, which is often used in Mobile phone and Notebook datasets are present:
„*Der Touchscreen ist ebenfalls alles andere als das Gelbe vom Ei."* (Mobile phone dataset).
*„Ich hätte manchmal große Lust, einen Hammer zu nehmen, um das Display zu bedienen". (*Mobile phone dataset).
„Recovery-CD/DVD muss selber gebrannt werden (hier beißt sich die Ratte in den eigenen Schwanz)." (Notebook dataset).
Also Finger Weg !!!! (Notebook dataset).
„Musikwiedergabe ohne Kopfhörer kann man getrost vergessen, da der Klang sehr blechern ist, als ob man über eine Konservendose Musik hören würde". (Mobile phone dataset)
Das mitgelieferte Headset ist allerdings nicht so der Hit. (Mobile phone dataset)
In the following some examples of negation formulated in subjunctive form are presented
„Man würde sich wohl im Display die Nase pudern können, einen Pickel ausdrücken können (natürlich ein wenig übertrieben) - aber nun mal ehrlich: für den Ausseneinsatz ist dieses Gerät nicht wirklich geeignet." (Notebook dataset).
„Die Sprachqualität könnte besser sein." (Mobile phone dataset)
„Ich hätte mir eine Höhere Auflösung gewünscht." (Mobile phone)
„Auch wäre ein Capslock wünschenswert gewesen, denn ich schreibe öfter mal ganze Worte groß und bei gleichzeitig gedrückter Shifttaste ist das ab und wann reine Fingerakrobatik." (Notebook dataset)
„So vermisse ich etwas den glänzenden, schwarzen Rand um das Display, welcher mir bei den anderen Apple Computern sehr gut gefällt, ab und zu wäre eine beleuchtete Tastatur auch sehr wünschenswert und ein integrierter UMTS Chip würde dem Gerät auch gut zu Gesicht stehen." (Notebook dataset).

Appendix C

List of polarity words from Notebook and Mobile phone datasets, which are not present in SentiWS dictionary.

Words with negative polarity	Words with positive polarity
Inkompabilität, Mängel, Mangel, gewöhnungsbedürftig, Minuspunkt, bemängeln, ärgert, fehlende, Totalverfehlung, blassen, Verbesserungspotential, stinkig, Contra, Kritikpunkt, unangebracht, Blechern, unbrauchbares, kotzen, arg, argen, unerreichbar, automatisch, warnen, Manko, Mankos, wacklig, lieblos, Steifheit, Nervfaktor, verbaute, verbauen, minderwertig, nervig , umständlich, Qual, laut, lahm, Nachteil, leistungsschwache, leistungsschwach, Unding, nervt, bemängeln, mühsam.	erreichbar, geschützt, wahrnehmbar, automatisch, nativ, passenden, erforderlich, echtes, Synchronisation, wertig, höherwertigen, höherwertig, geeignet, eignen, klappen, höherwertigen, ausgereift, vorgesehen.

Table C.1: List of sentiment words from Mobile phone and Notebook datasets not present in SentiWS.

Bibliography

- David Joshua Perdue, Social Media Marketing: Gaining a Competitive Advantage by Reaching the Masses, Spring 2010.
- Tom Smith, Power to the people Social Media Tracker Wave 3, 2008.
- Bing Liu, Web Data Mining, Berlin 2007.
- Bing Liu, *Sentiment Analysis and Subjectivity.* 2010.
- Pang, Bo, and Lee Lillian. *Opinion Mining and Sentiment Analysis.* 2008.
- Alistair Kennedy and Diana Inkpen, Sentiment classification of movie reviews using contextual valence shifters, 2006.
- Ellen Riloff and Janyce Wiebe, Learning extraction patterns for subjective expressions. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2003.
- Janyce M. Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin, Learning subjective language. *Computational Linguistics*, 30(3):277–308, September 2004.
- Urbansky David, Muthmann Klemens, Philipp Katz, Reichert Sandro, TUD Palladian Overview, April 2011.
- Benjamin Snyder and Regina Barzilay. Multiple aspect ranking using the Good Grief algorithm. In *Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL)*, pages 300–307, 2007.
- Sanjiv Das and Mike Chen. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific Finance Association Annual Conference (APFA)*, 2001.
- Van Rijsbergen, *Information Retrieval*, 1979.
- C. Fellbaum, ed., Wordnet: An Electronic Lexical Database. MIT Press, 1998.
- Esuli and F. Sebastiani, “SentiWordNet: A publicly available lexical resource for opinion mining,” Proceedings of Language Resources and Evaluation (LREC), 2006.
- Bas Heerschop, Alexander Hogenboom, and Flavius Frasincar, Sentiment lexicon creation from lexical resources, pp. 185-196, 2011
- Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Serach Engine. In: 7th International WWW Conference, pp. 107-117, 1998.
- Esuli and F. Sebastiani, “PageRanking WordNet synsets: An application to opinion mining,” Proceedings of the Association for Computational Linguistics (ACL), 2007.
- Robert Remus, Uwe Quasthoff, Gerhard Heyer, SentiWS – a Publicly Available German-language Resource for Sentiment Analysis, 2010.
- Waltinger, Ulli, GermanPolarityClues: A Lexical Resource for German Sentiment Analysis, 2010.
- K.W. Church and P. Hanks. 1990. Word Association Norms, Mutual Information, and Lexicography. Computational Linguistics, 16(1):22–29.
- Pang, B., Lee, L. Vaithyanathan, S., Thumbs up? Sentiment Classification Using Machine Learning Techniques, 2002.
- Bas Heerschop, Paul van Iterson, Alexander Hogenboom, Flavius Frasincar, and Uzay Kaymak, Analyzing Sentiment in a Large Set of Web Data while Accounting for Negation, 2010

- Kushal Dave, Steve Lawrence, David M. Pennock, Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews, In Proc. Of the 12th Intl. World Wide Web Conference (WWW'03), pp. 519-528, 2003.
- Benamara, F., Cesarano, C., Reforgatio, D.: Sentiment Analysis: Adjectives and Adverbs are better than Adjectives Alone. In: 1st International Conference on Weblogs and Social Media (ICWSM 2007), pp. 203–206. AAAI Press (2007).
- Peter D. Turney ,Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 417-424.
- Soo-Min Kim, Eduard Hovy, Determining the Sentiment of Opinions, in *Proceedings of the COLING conference, Geneva, 2004.*
- Cesarano, C., Dorr, B., Picariello, A., Reforgiato, D., Sagoff, A., Subrahmanian, V.: OASYS: An Opinion Analysis System. In: AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs (CAAW 2006). pp. 21–26, AAAI Press (2006)
- Ross, S.2001. A first course in probability, prenctice hall,2001.
- Wei Zhang, Clement Yu, UIC at TREC 2007 Blog Track, 2007.
- Lifeng Jia, Clement Yu, Wei Zhang, UIC at TREC 2008 Blog Track, 2008
- Breyten Ernsting, Wouter Weerkamp, Maarten de Rijke, Language Modeling Approaches to Blog Post and Feed Finding, 2007.
- Zdravko Markov, Daniel T. Larose, Data mining the Web, Uncovering Patterns in Web Content, Structure, and Usage, 2007
- Phayung Meesad, Pudsadee Boonrawd, and Vatinee Nuipian, A Chi-Square-Test for Word Importance Differentiation in Text Classification, 2011
- P. Saengsiri, P. Meesad, S. Na Wichian and U. Herwig, "Comparison of Hybrid Feature Selection Models on Gene Expression Data," IEEE International Conference on ICT and Knowledge Engineering, 2010, pp.13 -18.
- Sprejz, Michéle, Extraktion und Klassifikation von bewerteten Produktfeatures auf Webseiten, 2012
- Creative Research Systems, Survey Software Solutions. Available at http://www.surveysystem.com/sscalc.htm Last assessed on 12.01.2012 at 22:15:27.